NEC電子デバイス

μCOM-80トレーニング・キット
TK-80ユーザーズ・マニアル

NEC日本電気株式会社

IEM-560A
SEP-22-76

目　　次

第1章　概　　説

はじめに

ＴＫ－８０トレーニング・キットは，これからマイクロコンピュータを理解し，実際に使って見ようという方の便宜のために設計された８ビット・マイクロコンピュータのキットで，組み立てに必要な全部品と説明書が含まれています．従ってあなたは組み立て説明書の指示通りに部品を取り付けていけばキットを完成させることができます．

組み立てたセットは，プログラミング手段としてキー・ボード・スイッチと１６進数表示の８桁ディスプレイを備えていますので，電源を接続するだけですぐにプログラムを書き込んで，その場でそのプログラムを実行することができます．説明書の中には興味深いプログラム例が示されていますので，その通りプログラムしていけば楽しみながらプログラム・テクニックを少しずつ習得していくことができます．

このようにＴＫ－８０キットは，マイクロコンピュータのハードウェアとソフトウェアを具体的に体験しながら学ぶことができるため，極めて短時間でこの分野の知識を習得することができます．

このキットで採用している回路は，トレーニング・キットとしての機能を目的として低価格，簡略化を計っており，量産機種には不向きの部分もありますので，直接の採用はお控えください．なお当社は，特許権等に関する一切の責任を負いませんので，御了承ください．

注意　部品キットの開封は，マニアルの中で組み立ての指示があるまで行わないでください．

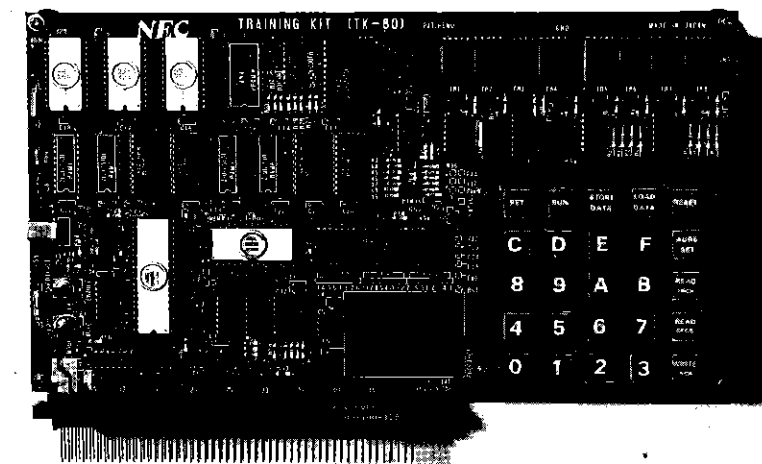
写真１－１　完成したTK－80キット

1.1 TK−80のシステム概要

図1−1 TK−80のシステム構成

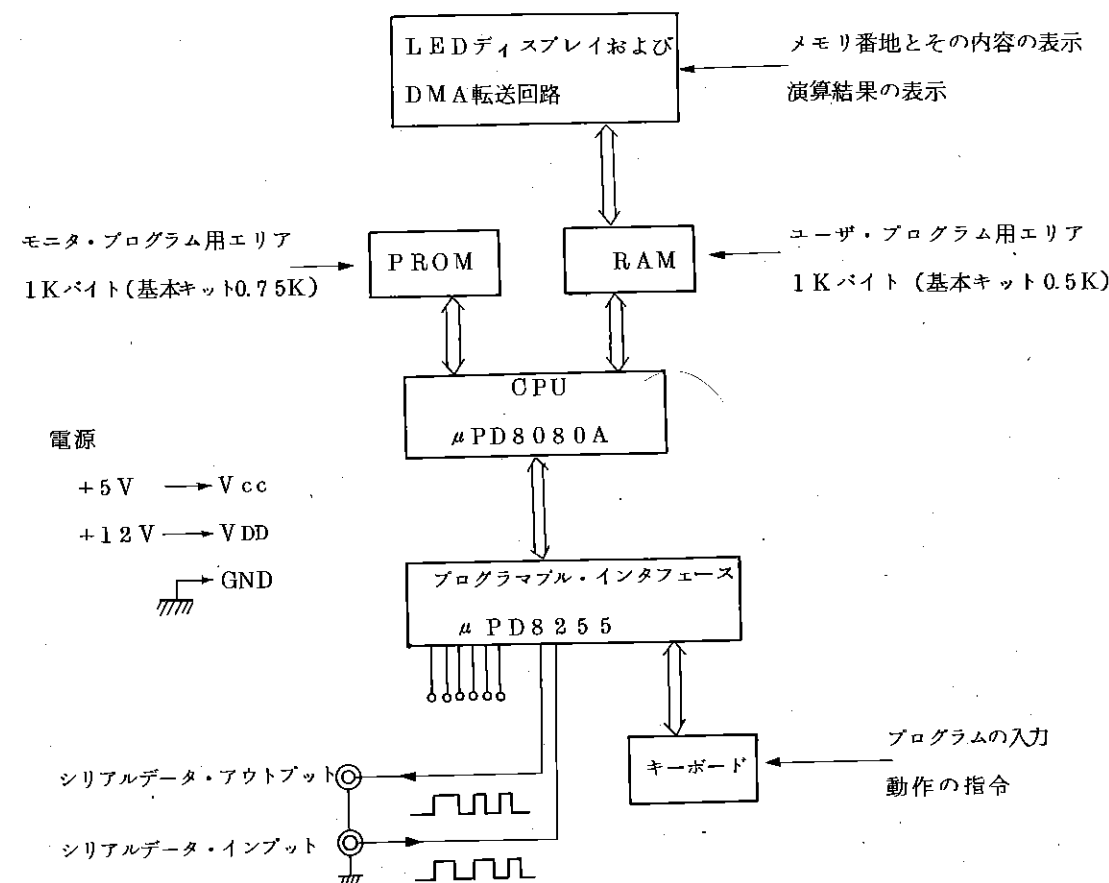



図1−1にTK−80のシステム構成を示します．このシステムは動作に必要な全回路を1枚のプリント・ボードに実装した極めてシンプルな構成になっています．このためボードへの部品取り付けが完了すると，外部より2種類の電源を供給するだけで，すぐにシステムを動作させることができます．

はじめにRESETキーを押すと，PROMに書き込まれているモニタ・プログラムが動作を開始します．モニタ・プログラムは，データや命令をキーボードからメモリに書き込んだり，読み出したりするためのプログラムです．この時メモリ番地とデータはそれぞれ16進数表現で，8桁の数字表示用LEDに表示されます．PROMを取り付けるスペースは4個分あり（PROM1個のサイズは256ワード×8ビットです），その内モニタ・プログラムに3個がすでに使われていますので，残り1個のスペースがユーザ・プログラム用として使用できます．

RAMはボード上に1,024バイトまで実装できますが（RAM 1個のサイズは256ワード×4ビット），最小構成では2チップ・256バイトで動作可能です．RAMを最小構成から順に増やして行く場合には，メモリ番地の大きい方から小さい方へという順番で実装して行きます．これはモニタ・プログラムで使用するスタック・エリアとワーキング・エリアがRAMの実装されるメモリ番地の最後にとられているためです．

TK−80で使用するRAMは，消費電流の非常に少ないCMOSですので，乾電池でバック・アップすれば他の電源を切っても，記憶した内容をそのまま保持することができます．

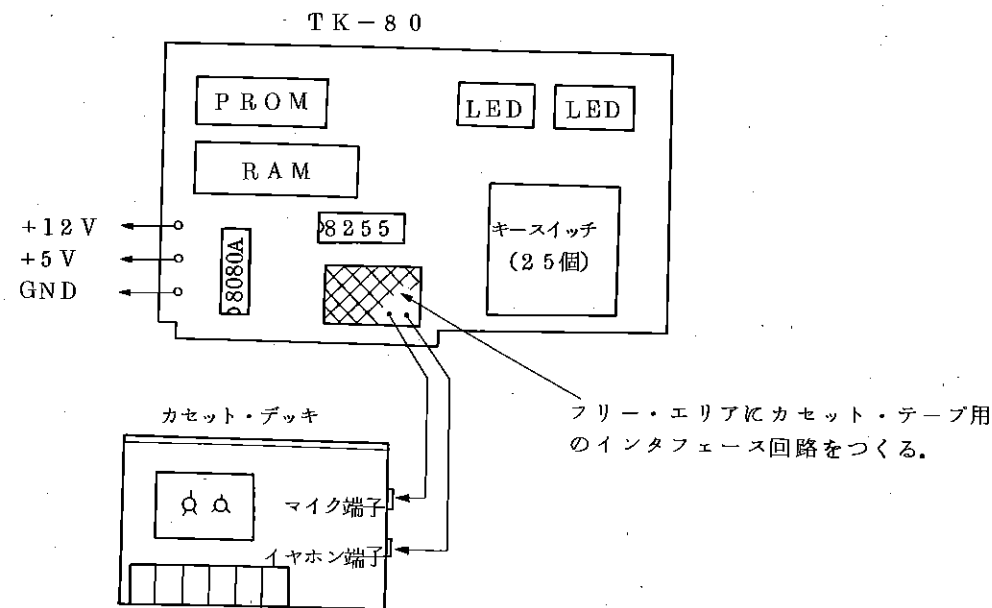
プログラマブル・ペリフェラル・インタフェースLSIは，キーボード回路のスキャニングに使用されます．また入出力ポートの内2本がシリアル・データ転送用に使用されます．シリアル・データとTK−80で使用されるパラレル・データとの変換は，モニタ・プログラムがソフトウェア上の処理で行っています．このシリアル・データ転送ラインにオーディオ周波数での変復調回路を追加すれば，簡単にオーディオ・テープ（カセット・テープが手軽です）を外部記憶装置として使用できます．

オーディオ・テープへの書き込み，読み出しのためのソフトウェアはモニタ・プログラムに含まれています．最も簡単な変復調回路は（第6章を参照してください），IC2〜3個とダイオード，抵抗，コンデンサおのおの数本で構成できますので，TK−80のプリント・ボードのフリー・エリアとして残されているスペースに組み込むことができます．

1.2 TK−80の仕様

CPU	μPD8080A	
クロック周波数	2.048MHz（18.432MHzクリスタル使用）	
ROM実装容量（MAX）	1,024バイト（ボード上MAX）	
RAM実装容量（MAX）	1,024バイト（ボード上MAX）	
パラレルI/Oポート	8ビット×3ポート（入力，出力はプログラム可能）	
入力装置	キーボード・スイッチ 25個（標準）	
表示装置	8桁7セグメントLEDによる16進数表示	
シリアルI/O	110ビット/秒のシリアル入・出力端子（シリアルデータのロード，ストア機能はモニタに含まれています）	
動作モード	シングルステップ，自動，をスイッチで切り換え	
RAMバックアップ	乾電池2本（3V）で可能	
電源	外部電源が必要	
	+5V ±5%	0.9A（MAX）
	+12V±5%	0.15A（MAX）
動作温度	0℃〜50℃	
寸法	310×180mm（プリント基板の寸法）	

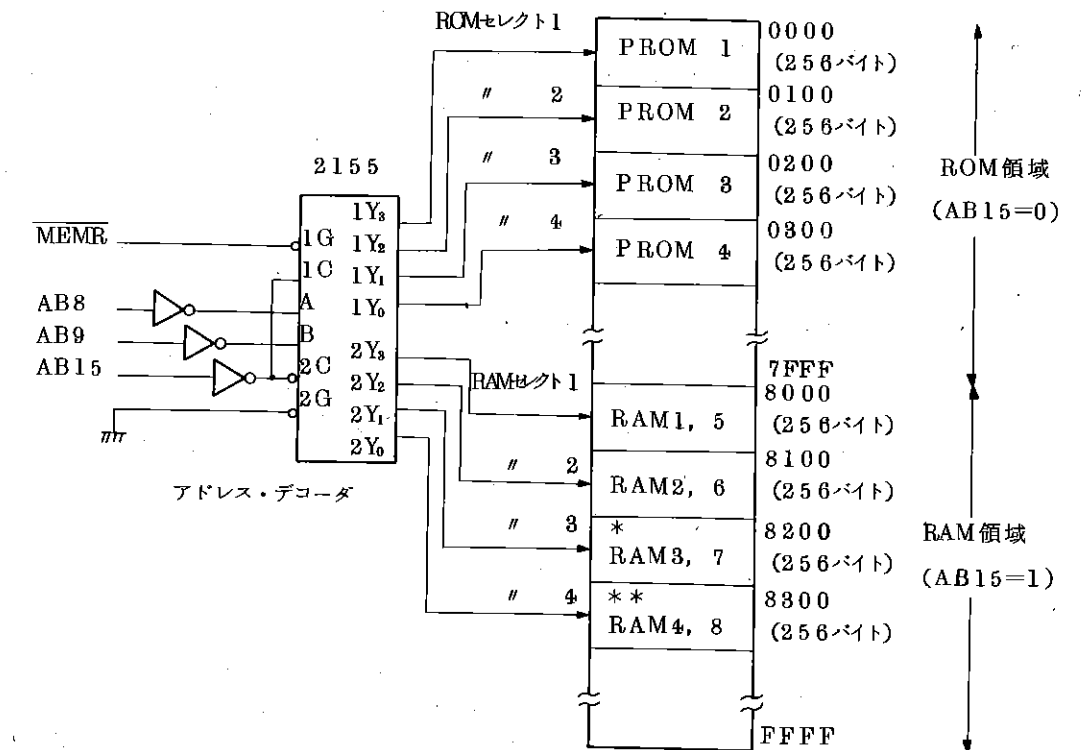
1.3　オーディオ・カセットの利用

図1－2　オーディオ・カセットの接続法

TK－80のモニタ・プログラムには，すでにオーディオ・カセットを外部記憶装置として利用するための，ロード，ストア機能が含まれています．実際には図1－2のような構成で利用できます．

1.4　システムの拡張性

システムに拡張性をもたせるために，プリント・ボードのコネクタ端子にアドレス・バスとデータ・バスが引き出されています．ただし，データ・バスはTTLでドライブ能力が強化されていますがアドレス・バスはCPU（MOS）から直接引き出されているので，大きな負荷が接続される場合には途中にバッファを追加する必要があります．

またメモリをボードの外に増設したい場合には，ボード上のメモリと追加したメモリを正しくアクセスさせるためのデコーダが必要です．このように適当なバッファやデコーダを追加すれば，システムは拡張することができます．

図1－3　基本構成のメモリ配置

** 最初にメモリを実装する場合は，RAM4，RAM8の位置に取り付けます（このエリアにスタックが確保されます）．

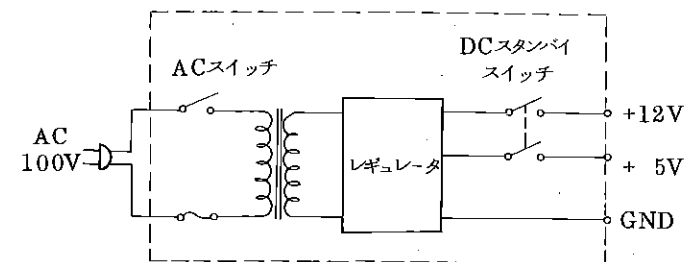
* 次に〔RAM3，RAM7〕，〔RAM2，RAM6〕，〔RAM1，RAM5〕の順に取り付けます．

1.5　電源に関する注意事項

ＴＫ－８０用の電源には，ＡＣスイッチとＤＣスタンバイ・スイッチの両方が付いているものを使用するようにして下さい。

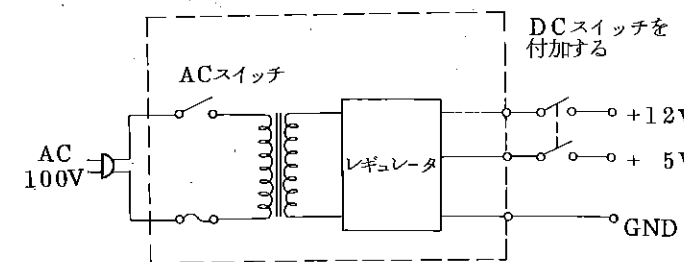
ＡＣスイッチだけが付いている電源を利用する場合は，ＤＣスイッチを外付けして，このスイッチで電源のオン／オフを行って下さい。

備考　一般の安定化電源では，ＡＣスイッチのオン／オフ時に電源トランスに発生したサージが直流出力にのって，スパイク状の異常電圧を誘起することがあります。
この異常電圧は，その程度によりますが最悪の場合にはＩＣなどの素子に悪影響を与えます。

(1)　好ましい電源

ＤＣ出力をＯＦＦにできるスイッチが付いている電源が好ましい。

投入順序
AC　SW　ON
↓
DC　SW　ON

（＋５Ｖ，＋１２Ｖのスイッチが独立している場合には＋５Ｖを先に投入して下さい）

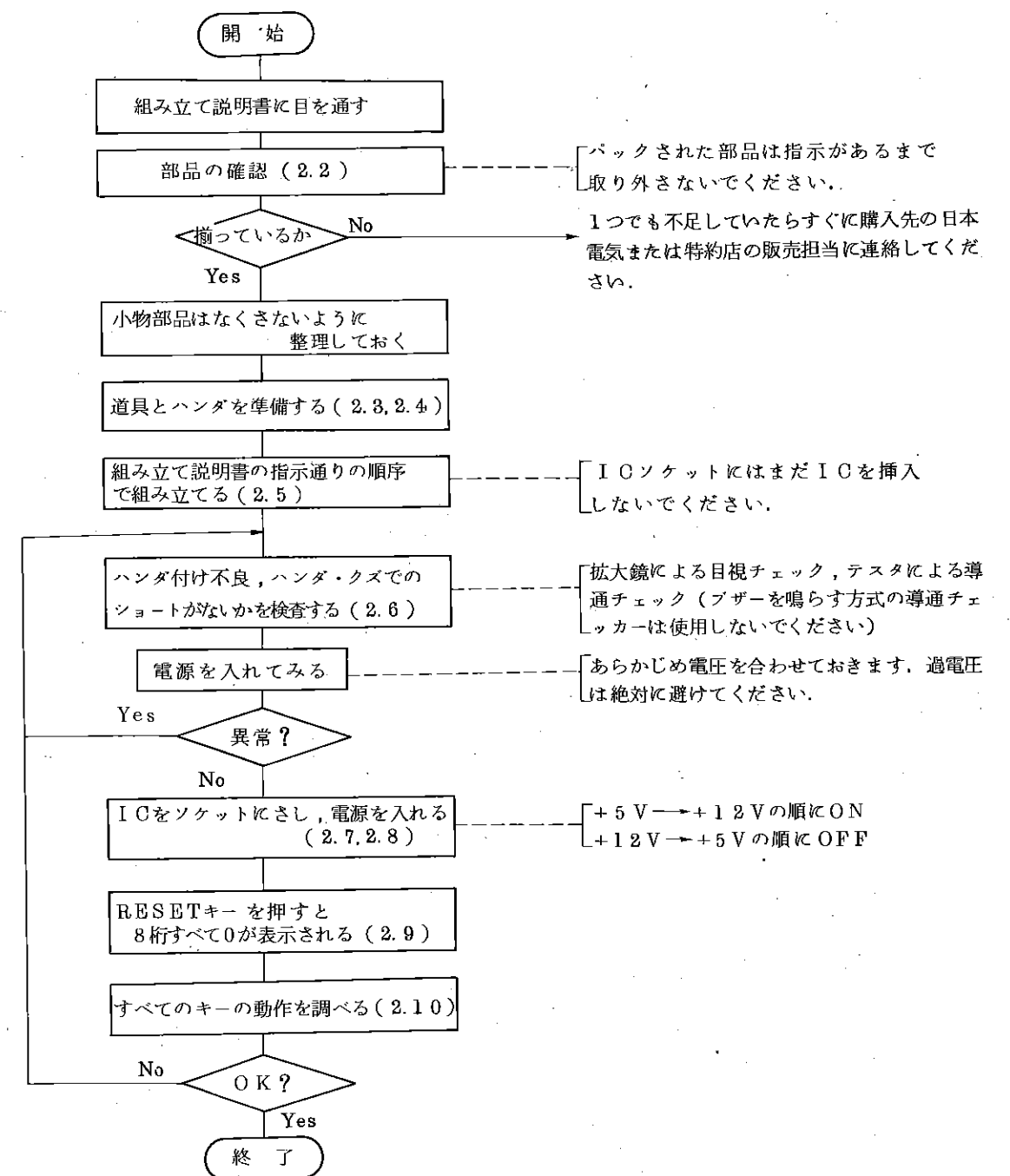
(2)　DCスイッチの外付け

切断順序
DC　SW　OFF
↓
AC　SW　OFF

（＋５Ｖ，＋１２Ｖのスイッチが独立している場合には＋１２Ｖを先に切断して下さい）

第２章　組み立て

2.1　組み立て作業の進め方

ＴＫ－８０キットの組み立ては，図２－１の順序で行ってください。

図２－１　組み立て作業の進め方

2.2 キット部品の確認

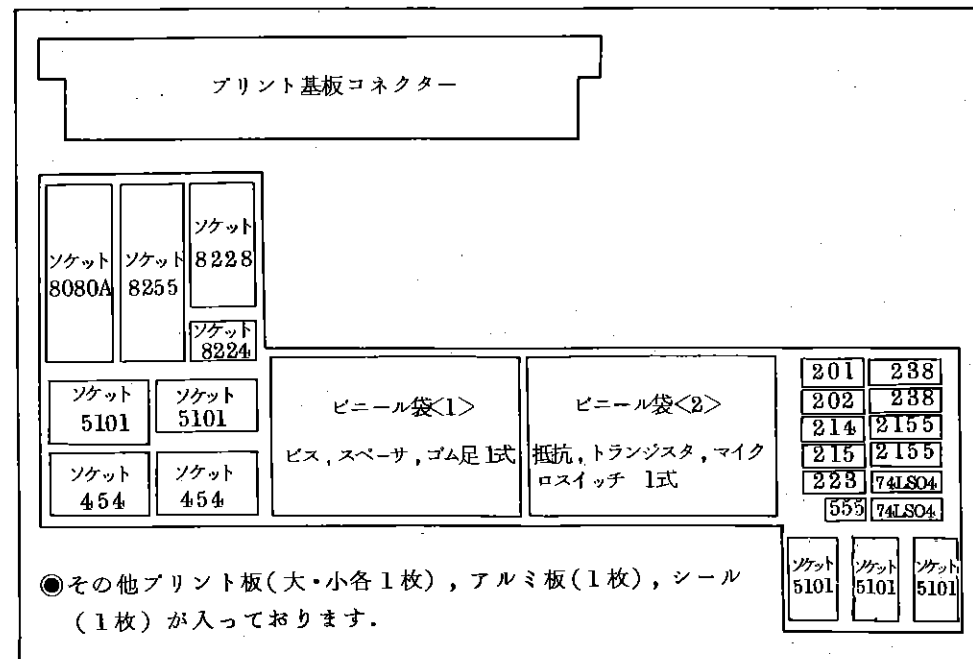
キットには，次の部品が含まれています．組立て前に必ず確認してください．

> 部品の中にはMOS LSIのように静電気に対して敏感な素子も含まれていますので，前面にパックされている部品は指示があるまで取付け台からはずさないでください．

(1) 前面にパックされている部品

IC	μPD8080A	×1	8ビット・パラレル・セントラル・プロセッサ・ユニット
	μPB8212	×1	8ビット I/Oポート
	μPB8224	×1	クロック・ジェネレータ・ドライバ
	μPB8228	×1	システムコントローラ・バスドライバ
	μPD8255	×1	プログラマブル周辺インタフェース
	μPD454	×3	フルデコード2048ビット EEPROM
	μPD5101	×4	フルデコード1024ビット スタティックRAM
LED	SN713A	×8	7セグメント固体発光数字表示素子
XTAL	18.432MHz	×1	HC18/Uタイプ水晶振動子
KEYスイッチ		×25	メカニカル接点キー・スイッチ

(2) 箱の内側に格納されている部品

図2－2 箱の内側の部品配置

IC	μPB201(7400)		×1
	μPB202(7410)		×1
	μPB214(7474)		×1
	μPB215(7401)		×1
	μPB223(7493A)		×1
	μPB238(7438)		×2
	μPB2155(74155)		×2
	74LS04		×2
	NE 555		×1
ICソケット	40 ピン		×2
	28 ピン		×1
	24 ピン		×3
	22 ピン		×4
	16 ピン		×1
プリント基板（大）			×1
プリント基板（小）			×1
プリント基板用コネクタ			×1
キー取付用アルミ板			×1
キー用文字シール			1組
ビニール袋(1)			
ビス			×7
ナット			×7
ワッシャ			×7
スペーサ（長）			×3
スペーサ（短）			×4
ゴム足			×7
スズメッキ線材			2m
エンパイア・チューブ			1m
ビニル線材			1m
ビニール袋(2)			
抵抗器	51Ω	1/4W	×8
	1KΩ	1/4W	×24
	5.1KΩ	1/4W	×8
	10KΩ	1/4W	×8
	15KΩ	1/4W	×8
	33KΩ	1/4W	×8
	51KΩ	1/4W	×2

コンデンサ	1μF	15WV	タンタル	×2
	10μF	25WV	タンタル	×1
	22μF	15WV	タンタル	×2
	0.01μF	50WV	セラミック	×34
トランジスタ	2SA713			×8
ダイオード	SD13			×4
	1S953／954			×2
トグルスイッチ	双極双投			×1
	単極双投			×1

確認の結果，1個でも不足していた場合は，購入先の日本電気または特約店の販売担当へ連絡してください．

写真2－1　TK－80キット

2.3　道具の準備

2.　3.　1　必要な道具，材料

○ハンダごて　○ニッパ　○テスタ　○ハンダ

ハンダごては，25W程度の小型で，こて先の細いものが必要です．古くなったこて先は新しいものと交換してください．ヒータとこて先の絶縁不良のものは絶対に使用しないでください．

ハンダは“ヤニ入りハンダ”を使用してください．ハンダの線径は0.7mm～1.0mmのものが適当です．太すぎますとハンダが多く，盛り上がり過ぎて隣りのパターンとのショートの原因になりやすいのでよくありません．

2.　3.　2　あると便利な道具

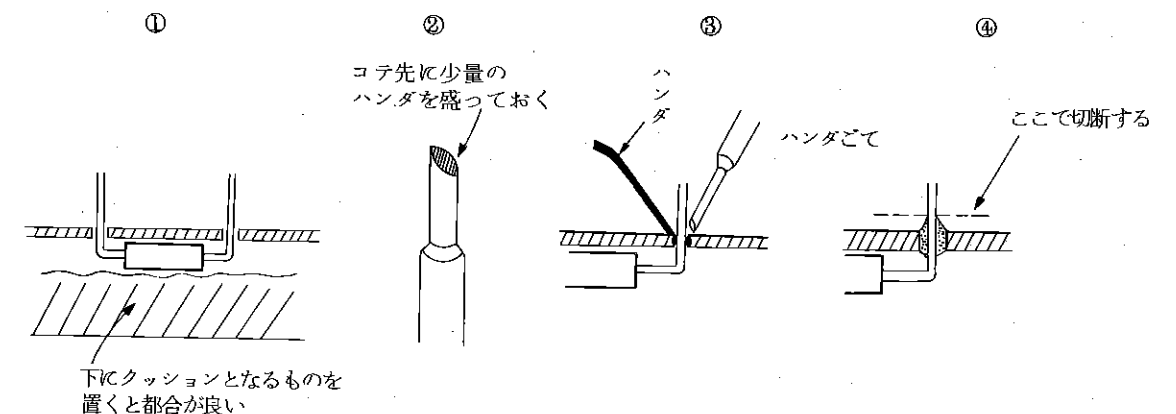
○拡大鏡　○十字ドライバ　○先き細ペンチ

○ハンダ吸い取り用のフラックス含有編線

拡大鏡はハンダ付け完了後，配線パターンが“ハンダくず”や“細いハンダの糸”でショートしていないかを調べるのに便利です．虫めがね，ルーペなどなんでも結構ですが，あると重宝なものです．

2.4　ハンダ付けに関する注意

ハンダ付けの前にハンダごての手入れを行うように心掛けてください．こて先の太いものや古くなってボロボロになっているものは不適当です．ヤスリで磨くか新しいものと交換してください．またこて先とヒータが絶縁不良になっていますと，半導体部品を破壊する恐れがありますので，一度テスタで絶縁状態を調べてください．

図2－3　ハンダ付け手順

ハンダ付け作業は次の手順で行います.

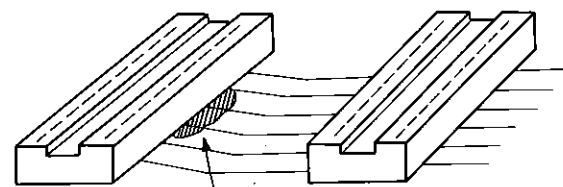
(1) まず部品を取り付け穴に差し込み，裏返しにします．この時クッションとなるものを下に置くと部品がしっかり押え付けられます.

(2) ハンダごてのこて先に少量のハンダを付け，ハンダになじませて置きます.

(3) ハンダごてをハンダ付けする場所とリード線に密着するようにあてがって加熱します．ほぼ同時か少し遅れてハンダ線をこの場所にあてがうとハンダが溶け始めます．ハンダがスルーホールの穴に吸い込まれ，接合部に行きわたったらハンダとハンダごてを離します.

(4) 不要のリード線をニッパで切り捨てればハンダ付けは完了します．半導体部品は長時間過熱することはできませんので，この作業は2～3秒程度で終わるようにしてください.

注 ハンダは多く盛り過ぎないように注意してください．余分のハンダは隣りのパターンへ流れたり，思わぬ障害になることがあります．特にICソケットを取り付ける際には，できるだけ少量のハンダで済ませてください．プリントの穴はスルーホールですので，ハンダは片面だけで良く裏面までしみ込ませる必要はありません．あまりハンダを流し込み過ぎますと，ソケットに隠れて見えない部分でピン間のショートを起こ[注(1)]すことがありますので注意してください.

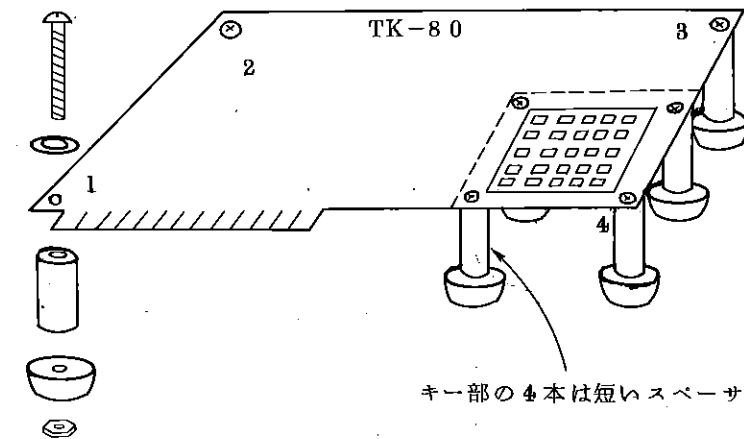
また，プリント・ボードの上で不用意にハンダごてを振るのは避けてください．落ちたハンダが思わぬ場所に入り込んで気が付かないことがありますので注意してください.

注(1) このキットではプリント・ボードのハンダ部分以外は，レジストでカバーされていますので，ブリッジは起こりにくくなっていますが念には念を入れてください.

2.5 組み立て

2. 5. 1 スペーサとアルミ・ボードの取り付け

プリント・ボードの下面にスペーサとキー取り付け用アルミ・ボードを取り付けます.

キーはまだ取り付けないでください.

図2－4 スペーサとアルミ・ボードの取り付け

2. 5. 2 抵抗器の取り付け

抵抗はプリント・ボード上では，(a)のように白い線で表示されていますので，(b)のように白い線の両端の穴に取り付けてください.

また，(c)のように線の途中に穴がある場合は無視して，(d)のようにやはり両端の穴に取付けてください.

抵抗は全部で66個あります．R1から順にR66まで表2－1に示す値に応じて取り付けてください.

(a) R15
(b) R15
(c) R16
(d) R16

プリント・ボード上の取り付け位置は，付図(I)を参考にしてください．取り付けの終わった部分は色鉛筆で，(R15)のように塗りつぶしておくと間違いがありません.

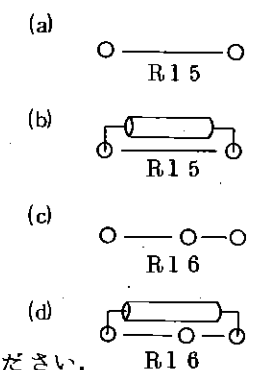



表2－1　抵抗器部品表

番号	抵抗値	カラーコード	番号	抵抗値	カラーコード
R1	5.1 KΩ	緑・茶・赤	R34	33 KΩ	橙・橙・橙
2	10 KΩ	茶・黒・橙	35	33 KΩ	〃
3	5.1 KΩ		36	33 KΩ	〃
4	10 KΩ		37	33 KΩ	〃
5	5.1 KΩ		38	33 KΩ	〃
6	10 KΩ		39	33 KΩ	〃
7	5.1 KΩ		40	33 KΩ	〃
8	10 KΩ		41	33 KΩ	〃
9	5.1 KΩ		42	15 KΩ	茶・緑・橙
10	10 KΩ		43	15 KΩ	〃
11	5.1 KΩ		44	15 KΩ	〃
12	10 KΩ		45	15 KΩ	〃
13	5.1 KΩ		46	15 KΩ	〃
14	10 KΩ		47	15 KΩ	〃
15	5.1 KΩ		48	15 KΩ	〃
16	10 KΩ		49	15 KΩ	〃
17	1 KΩ	茶・黒・赤	50	1 KΩ	茶・黒・赤
18	51 Ω	緑・茶・黒	51	51 KΩ	緑・茶・橙
19	51 Ω	〃	52	1 KΩ	茶・黒・赤
20	51 Ω	〃	53	1 KΩ	〃
21	51 Ω	〃	54	51 KΩ	緑・茶・橙
22	51 Ω	〃	55	1 KΩ	茶・黒・赤
23	51 Ω	〃	56	1 KΩ	〃
24	51 Ω	〃	57	1 KΩ	〃
25	51 Ω	〃	58	1 KΩ	〃
26	1 KΩ	茶・黒・赤	59	1 KΩ	〃
27	1 KΩ	〃	60	1 KΩ	〃
28	1 KΩ	〃	61	1 KΩ	〃
29	1 KΩ	〃	* 62	1 KΩ	〃
30	1 KΩ	〃	63	1 KΩ	〃
31	1 KΩ	〃	64	1 KΩ	〃
32	1 KΩ	〃	65	1 KΩ	〃
33	1 KΩ	〃	66	1 KΩ	〃

備考　抵抗器はすべて¼W，許容差±10％です．

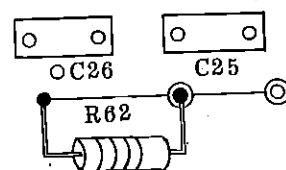
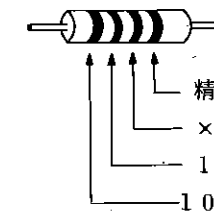
＊　R62の取り付け位置は下図のようになります．間違わないよう注意してください．

2.5.3　ダイオードの取り付け

表2－2　ダイオード

番号	品名	規格
D1	ゲルマニウム・ダイオード	SD 13
2	〃	〃
3	シリコン・ダイオード	1S953/954
4	ゲルマニウム・ダイオード	SD 13
5	〃	〃
6	シリコン・ダイオード	1S953/954

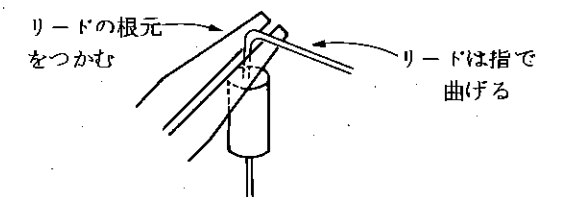
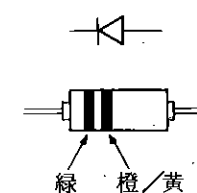
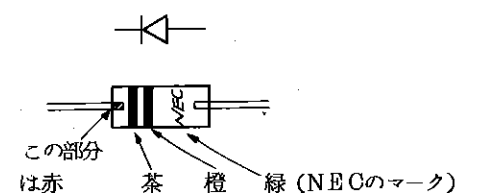
ダイオードはすべてガラス封入タイプですので，リード線を曲げる時，ガラスに力が加わらないように注意してください．

ダイオードには図2－5のように，品名を示す2桁のカラーコードが付いています．極性はこのカラーコードの付いている方がカソード側ですので，間違わないように取り付けてください．

図2－5　ダイオードの品名および極性表示

1S953/954の表示　　SD13の表示

2. 5. 4　コンデンサの取り付け

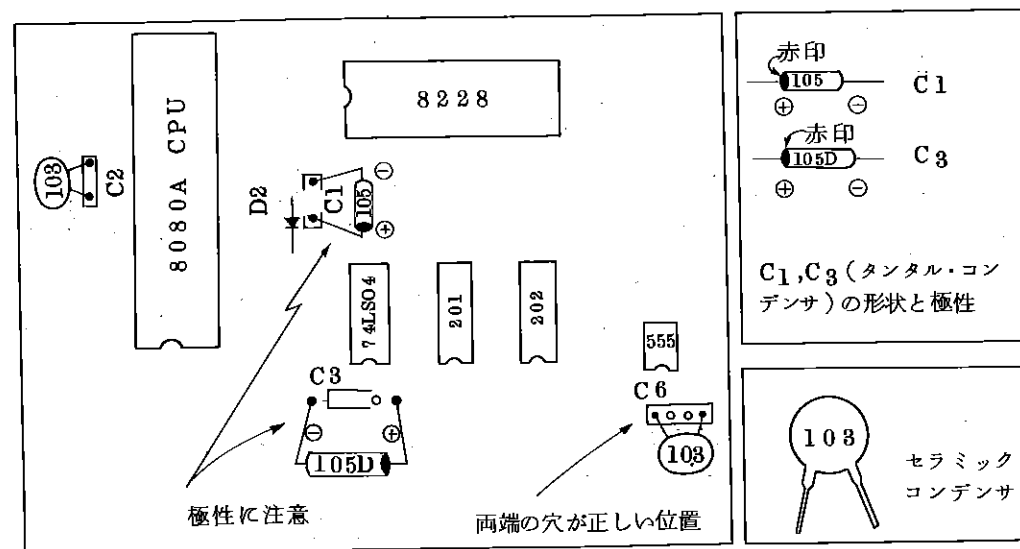
コンデンサは全部で３９個あります．その内Ｃ１，Ｃ２，Ｃ３，Ｃ６は回路の動作上，絶対必要なものであり，それ以外は電源のバイパス・コンデンサとして使われます．

Ｃ１，Ｃ２，Ｃ３，Ｃ６を最初に取り付ければ確実です．

表２－３　最初に取り付けるコンデンサ

番　号	品　　名	規　　格
Ｃ１	タンタル・コンデンサ	１μＦ　１５ＷＶ
２	セラミック・コンデンサ	０.０１μＦ　２５ＷＶ
３	タンタル・コンデンサ	１μＦ　１５ＷＶ
６	セラミック・コンデンサ	０.０１μＦ　２５ＷＶ

注　Ｃ１，Ｃ３はタンタル・コンデンサで有極性ですので，プリント基板上には図２－６の極性となるように取り付けてください．

図２－６　Ｃ１，Ｃ３の取り付け方向

次にＣ３７，Ｃ３８，Ｃ３９を取り付けます．この３つもタンタル・コンデンサですので極性には十分注意してください．

プリント基板上には，o-[⊕]-o　または　o-[　]-o+　のマークが付いていますので，⊕または＋マークの付いている方にプラス側を接続してください．

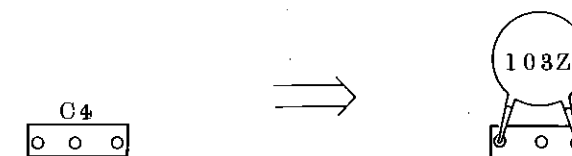
表２－４　極性に注意するコンデンサ

番　号	品　　名	規　　格
Ｃ１	タンタル・コンデンサ	１μＦ　１５ ＷＶ
３	〃　　〃	１μＦ　１５ ＷＶ
３７	〃　　〃	２２μＦ　１５ ＷＶ
３８	〃　　〃	１０μＦ　２５ ＷＶ
３９	〃　　〃	２２μＦ　１５ ＷＶ

最後にＣ４，Ｃ５，Ｃ７～Ｃ３６を取り付けます．これらはすべて電源バイパス用コンデンサです．極性はありません．

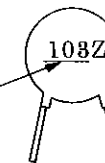
表２－５　バイパス用コンデンサ

番　号	品　　名	規　　格
Ｃ４	セラミック・コンデンサ	０.０１μＦ　５０ＷＶ
５	〃　　〃	０.０１μＦ　５０ＷＶ
７～３６	〃　　〃	０.０１μＦ　５０ＷＶ

コンデンサの取り付け位置を示す記号で，マークの中に２個以上の穴がある場合は，両端の２つの穴が正しい取り付け穴です．

＊セラミック・コンデンサの形状

103は，10×10^3 PF

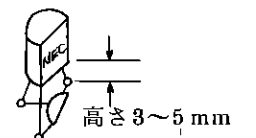
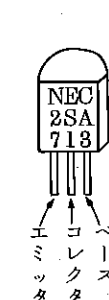
すなわち０.０１μＦを示します．

103Z

2. 5. 5　トランジスタの取り付け

トランジスタはＬＥＤのドライブ用に８個使用します．プリント基板には -◁ のようにマークされていますので，真上から見てマークとトランジスタの外形が一致するように取り付けてください．

表２－６　トランジスタ

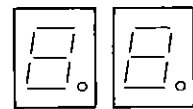
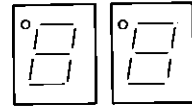
番　号	品　　名	規　　格
TR1～8	PNP ダーリントン・トランジスタ	２ＳＡ７１３

取り付けの高さは低い方が安定して良いのですが，リードを無理に変形させない程度（３mm位）としてください．

2. 5. 6 ＬＥＤの取り付け

ＬＥＤは８個あります．それぞれ位置がずれないよう，同じ高さに取り付けてください．

表２－７　ＬＥＤ．

番　号	品　　名	規　　格
LED1～8	７セグメントＬＥＤ	ＳＮ７１３Ａ

図２－７　ＬＥＤの取り付け方

○正しい取り付け方向　　×誤った取り付け方向

小数点が下側（トランジスタの並ぶ側）となるように取り付けます．

2. 5. 7 ＩＣの取り付け

ソケットを使用しないＩＣは，プリント基板に直接ハンダ付けします．

表２－８　ハンダ付けするＩＣ

番　号	品　　名	機　　能
IC１	ＳＮ７４ＬＳ０４	Hex Inverter
２	μＰＢ　２０１　(７４００)	Quad 2-Input NAND
３	μＰＢ　２１４　(７４７４)	Dual D Flip-Flop
４	μＰＢ　２０２　(７４１０)	Trip 3-Input NAND
５	μＰＢ　２１５　(７４０１)	Quad 2-Input NAND O/C
６	μＰＢ　２２３　(７４９３Ａ)	4-Bit Binary Counter
７	μＰＢ　２１５５(７４１５５)	Dual 2-4 Decoder
８	μＰＢ　２１５５(７４１５５)	〃
９	ＳＮ７４ＬＳ０４	Hex Inverter
１０	μＰＢ　２３８　(７４３８)	Quad 2-Input NAND Buffer O/C
１１	μＰＢ　２３８　(７４３８)	〃
２８	μＰＢ　８２１２	8-Bit I/O Port
２９	ＮＥ　５５５	Timer

プリント基板上のＩＣ取り付け位置は，右図のようにマークされていますので，その番号と同じ品名のＩＣを取り付けてください．またＩＣパッケージの“くぼみ”の方向もマークと一致するように注意してください（・印は１ピンの位置を示します）．

２０１

ＩＣおよびＬＥＤの取り付け方向は，絶対に間違わないよう注意してください．

注　各ＩＣは最初に対角線上の２ピンだけをハンダ付けして，もう一度間違って取り付けていないか念を入れて確認し，その後で全部のピンをハンダ付けしてください．全ピン，ハンダ付け終了後ではきれいに取り外す事はまず期待できないと考えてください．

もし間違って取り付けていることがわかった場合は，ハンダ付けしたピンのハンダを完全に吸い取ってから軽くこじるようにして抜き取ってください．

ハンダを吸い取るためには，平編線にフラックスをしみ込ませたものが“SOLDER TAUL”の商品名で市販されていますので利用すると便利です．これを使ってきれいにハンダを吸い取るコツは，いつも編み線の新しい部分を使用し，ハンダがにじんできた部分はどんどん捨てていくことです．この方法は毛細管現象を利用して溶けたハンダを編み線に吸い込むもので，ハンダ除去の方法としては非常にすぐれています．

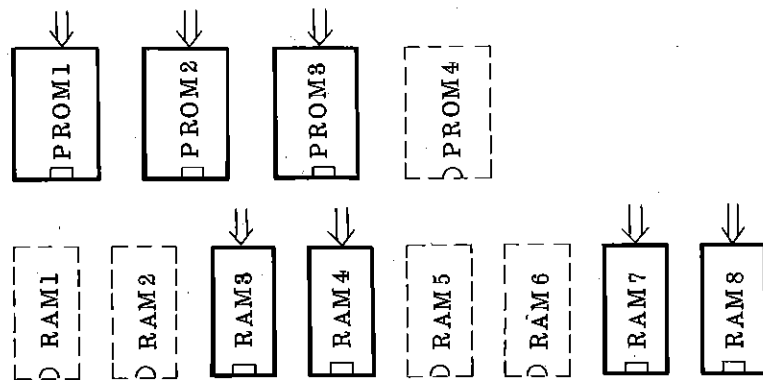
2. 5. 8 ＩＣソケットの取り付け

ＩＣソケットは１１個使用します．

表２－９　ＩＣソケット

ピン数	使用するＩＣ	
４０　ピン	μＰＤ８０８０Ａ	
４０　ピン	μＰＤ８２５５	
２８　ピン	μＰＢ８２２８	
２４　ピン	μＰＤ４５４	ＰＲＯＭ１
２４　ピン	μＰＤ４５４	ＰＲＯＭ２
２４　ピン	μＰＤ４５４	ＰＲＯＭ３
２２　ピン	μＰＤ５１０１	ＲＡＭ３
２２　ピン	μＰＤ５１０１	ＲＡＭ４
２２　ピン	μＰＤ５１０１	ＲＡＭ７
２２　ピン	μＰＤ５１０１	ＲＡＭ８
１６　ピン	μＰＢ８２２４	

ソケットのくぼみのある方向と，プリント基板上のＩＣマークのくぼみとを合わせて取り付けてください．最初に対角線上の２ピン（例えば１，４０ピン）をハンダ付けし，取り付けがゆがんでいないことを確かめてから，残りのピンを順番にハンダ付けしてください．

注意　２４ピン・ソケット３個は，ＰＲＯＭ１，２，３の指定位置に，また２２ピン・ソケット４個はＲＡＭ３，４，７，８の指定位置に取り付けてください．

ＩＣソケットとプリント基板との間にハンダが誤って入ってしまうと，ピン間のショートの原因となりやすく，しかも目視で見つけにくいものですから，くれぐれも注意してください．

ＩＣソケットのピン間隔（ピッチ）は２.５４mm（0.1インチ）と狭いので，できるだけ細いハンダ線を使用し，ハンダを盛り過ぎないようにしてください．

キットに含まれているＩＣソケットはすべてハンダ・ディップ用で足の短いものです．このため，動作時にＬＳＩの各端子での波形を観測する必要のある場合は，あらかじめラッピング・タイプのＩＣソケットを取り付けておく方が便利です，ソケットの足が長いのでクリップなどによる信号線の引き出しが容易となります．

2. 5. 9　水晶振動子の取り付け

水晶振動子は右図のようにリード線を曲げて取り付けます．

この振動子のケースはＨＣ１８／Ｕタイプと呼ばれる小型のものですので，通常の利用状態ではリード線による支持だけで充分ですが，振動の多い状況での利用では，プリント基板に接着してしまうことをおすすめします．

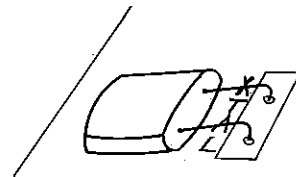



表２－１０　水晶振動子

品　名	規　　格
ＸＴＡＬ	基本発振モード　１８.４３２MHz　＊

＊　振動子のケースには周波数が刻印されていませんが，正確に１８.４３２MHzで発振するものを出荷しております．

2. 5. 10　トグル・スイッチの取り付け

トグル・スイッチは２個取り付けます．

図２－８の取付け図のように取り付け，配線（ビニル線）は裏面で配線図通り行ってください．

表２－１１　トグル・スイッチ

番　号	品　　名	規　格
ＳＷ１	トグル・スイッチ	単極双投
２	〃	双極双投

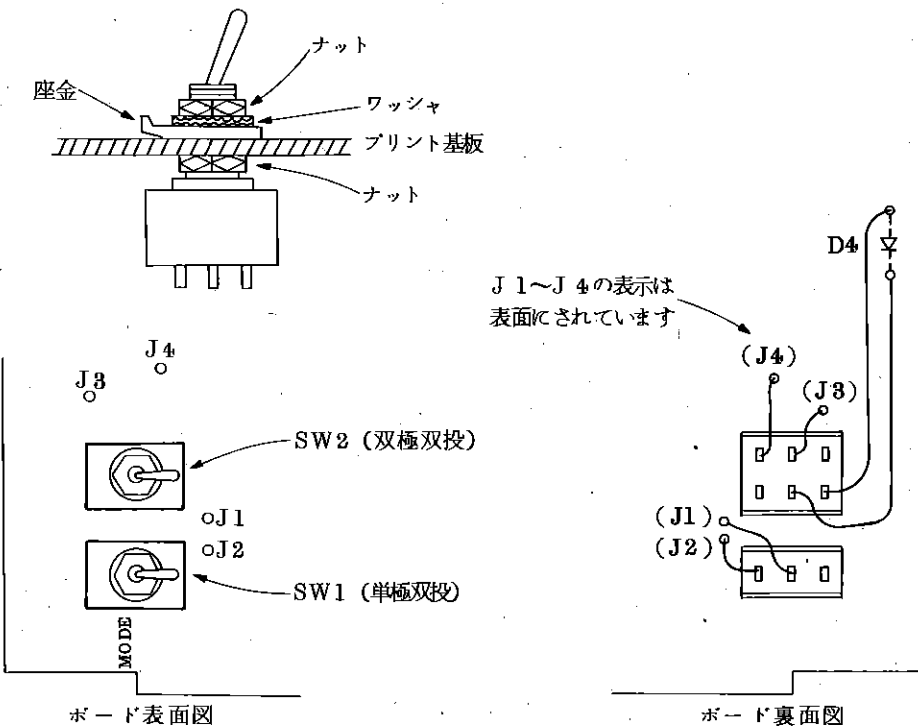
図２－８　トグル・スイッチの取り付け

2. 5. 11 キー・スイッチの取り付けおよび配線

表２－１２　キー・スイッチおよび取り付け用部品

品　名	個　数	規　格
キー・スイッチ	２５	メカニカル接点型
アルミ・ボード	１	キー取付け板
文字シール	１	キー用文字
スズメッキ線	２ m	共通ライン配線用
ビニル線	１ m	キー←→プリント・ボード配線用
エンパイア・チューブ	１ m	メッキ線用カバー

(1) キー・スイッチの取り付け

キー・スイッチは２５個あります．キーは１個ずつの独立型ですので，取付け用アルミ・ボードに取り付けて使用します．キーの文字は付属の文字シールをはがし，キーの透明キャップをはずして貼り付けてください．

キー・スイッチの取り付けは，写真２－２，図２－９，図２－10を参照して行ってください．

> 各キーの端子の方向は，図２－９の裏面配置図に従って配置すると配線が楽です．
> 文字シールは配線を行う前に貼りつけてください（キーが並んでしまうとキャップが取りはずせなくなります）．

キーの並べ方はこの通りにしてください（上より見た図）．

写真２－２　キー配置

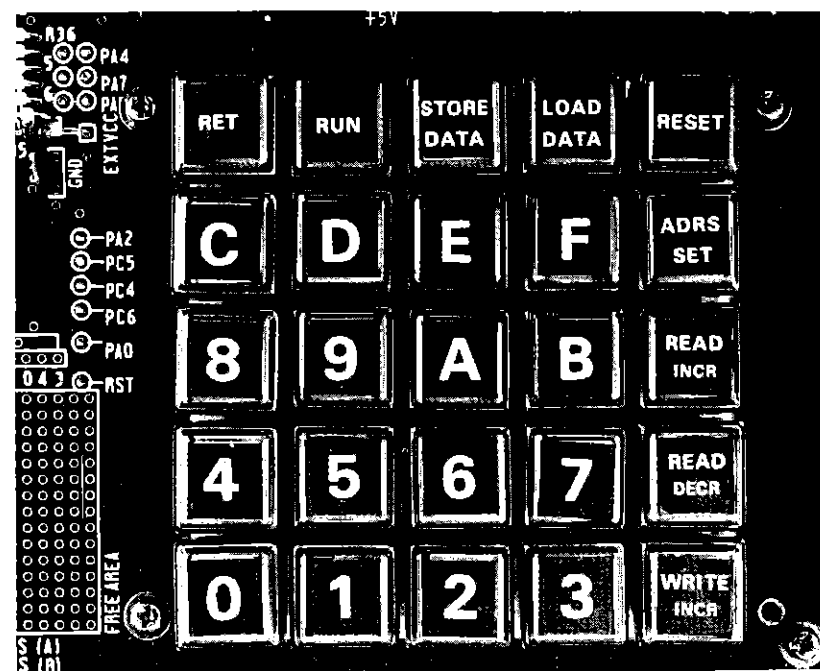



図２－９　キー配置図（裏面）

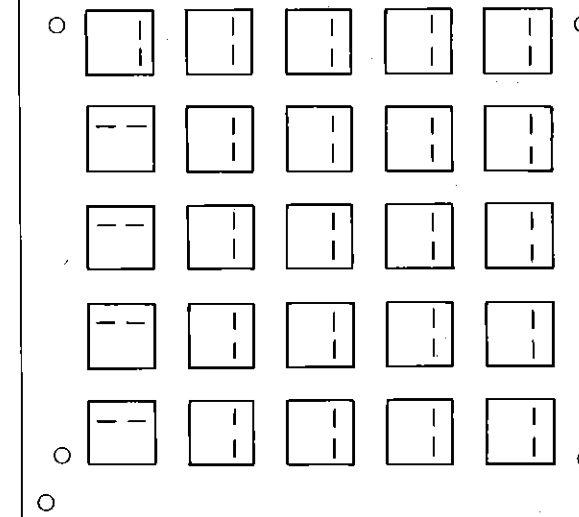
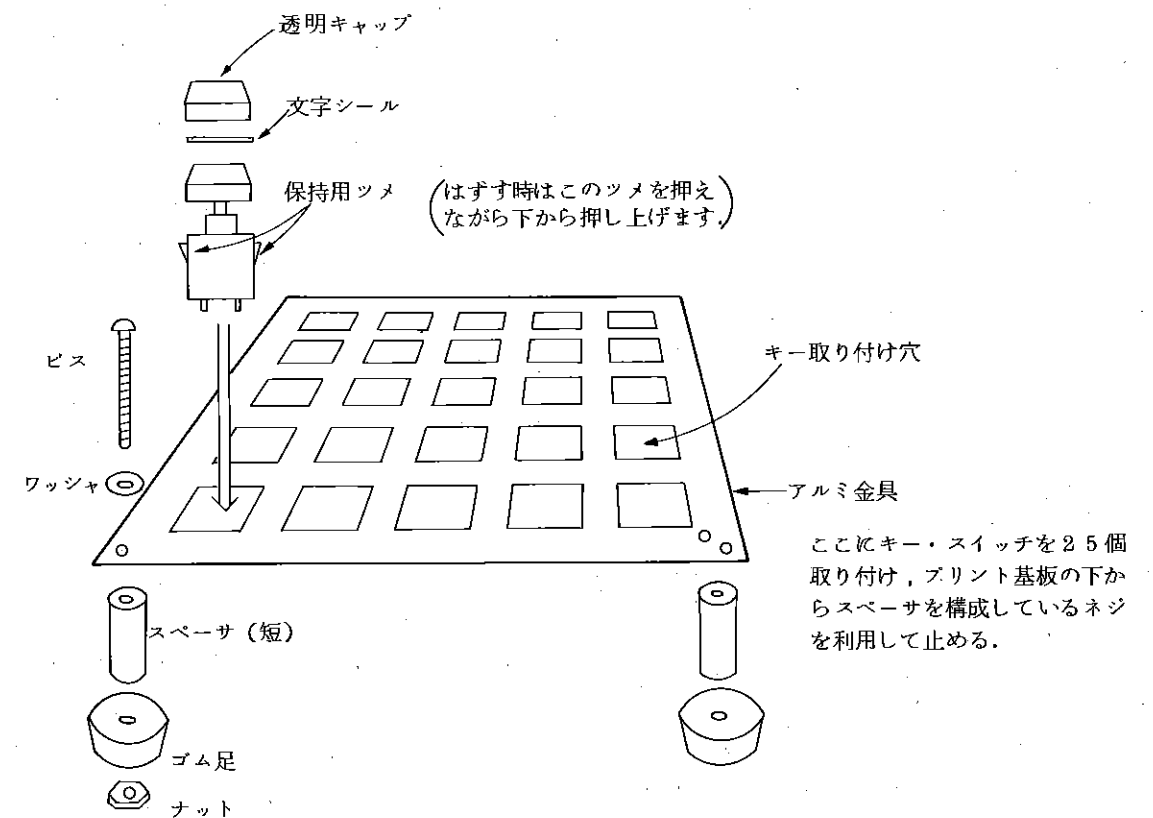



図２－１０　キーの取り付け方法

(2) キー・スイッチの配線

キーの配線はキットに含まれるスズメッキ線とビニル線を使用し，図２－１１および図２－１２に従って行います．

メッキ線の交叉する箇所にはエンパイア・チューブをかぶせてください．

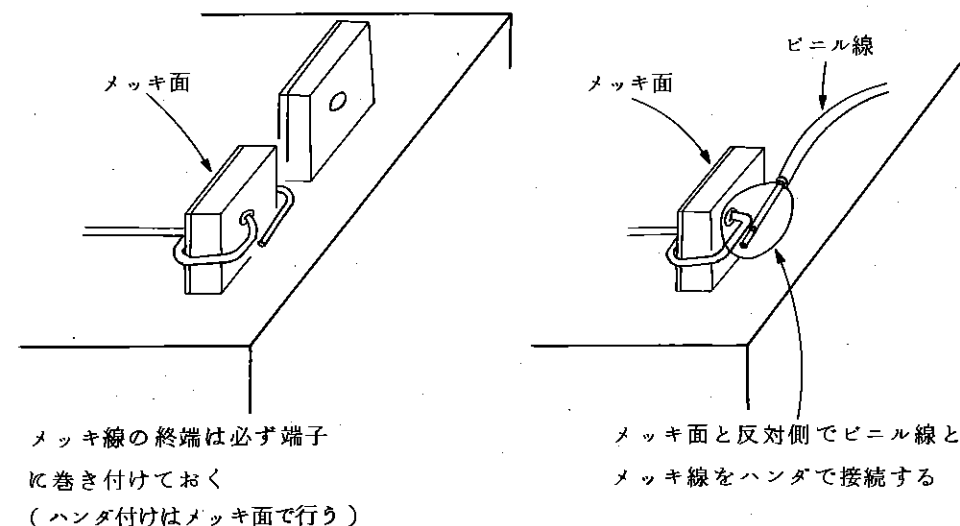
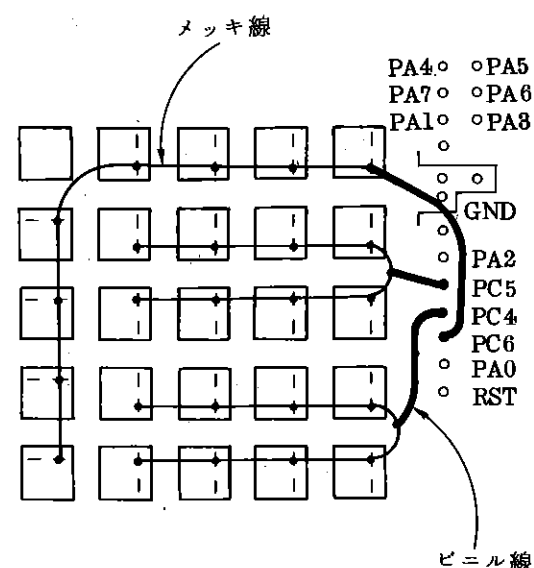
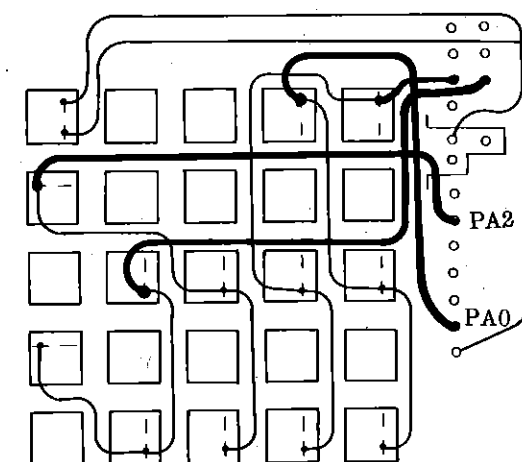
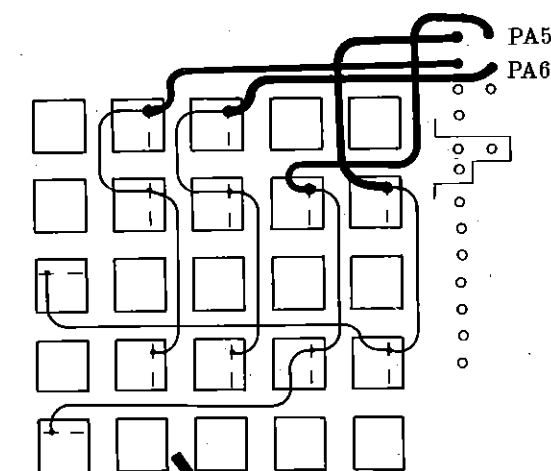
図２－１１　キー・スイッチの端子配線

メッキ面の引張りに対する強度はそれ程大きくありませんので，メッキ面に直接引き出し線をハンダ付けすることは避けてください．

図２－１２　キー・スイッチの配線（裏面）

①裏面より見てキーの端子の並びを図のようにする．

②メッキ線の共通ライン（それぞれ８個の端子を通る）を端子の穴を貫いてつくる．

③３本の共通ラインをそれぞれＰＣ４，ＰＣ５，ＰＣ６へビニル線で接続する．

④同様に共通ラインを４本のメッキ線でつくる．

⑤その先端をそれぞれＰＡ０，ＰＡ１，ＰＡ２，ＰＡ３へビニル線で接続する．

⑥ＲＥＳＥＴキーの２本はそれぞれメッキ線（チューブをかぶせる）でＲＳＴ，ＧＮＤへ接続する．

⑦更に４本の共通ラインをメッキ線でつくる．

⑧その先端をそれぞれＰＡ４，ＰＡ５，ＰＡ６，ＰＡ７へビニル線で接続する．

2.6 検査

ハンダ付け作業が完了すると，ＩＣソケットにＩＣを取り付ける前に配線の状態を目視で検査します．

特に注意すべき点は，“ハンダくず”や“ハンダ糸”によるパターン間のショートです．パターンの間隔は狭いので肉眼では見落とす恐れがあります．このような場合，拡大鏡があると便利です．

また，半導体部品を実装した後での検査では，導通試験器（例えば直接ブザーを鳴らすタイプ）の使用は避けてください．テスタを使用することをおすすめします．

2.7 ICソケットへのICの実装

ＭＯＳ　ＩＣは，静電気による異常な高圧が入力端子（ゲート）に加わりますと，破壊する恐れがありますので，注意して取り扱ってください．

静電破壊に対する対策の基本には次の３つが上げられます．

(1) 静電気を発生させないこと

(2) 発生した静電気は逃がしてやること

(3) ICと接触する物体はあらかじめ同電位にしておくこと

(1)のためには静電気を発生しやすいものを身の回りに置かないことと，机の上をあらかじめ濡れ雑巾で拭いて置くと効果があります．(2)(3)のルールを守る簡単な方法は手で触れることです．例えばICをソケットに挿入する前に，ICとソケットの両方を手で触って置けば同電位になります．

ただし，人体自身が帯電しているとかえって悪影響をおよぼしますので，アースに対して少しでもリークしやすい物に触れてから行ってください．ジュータンの上をスリッパで歩いた後は，特にこの点に注意してください．

パックからMOS IC を抜き取る時は，必ずアルミ・シートに手で触れてからにしてください． またソケットに挿入する場合も，その前にソケットの足に手で触れてください． ソケットからMOS ICを抜いて他の場所へ置く場合には，置く場所（例えば銀紙の上）にまず手を触れてください．

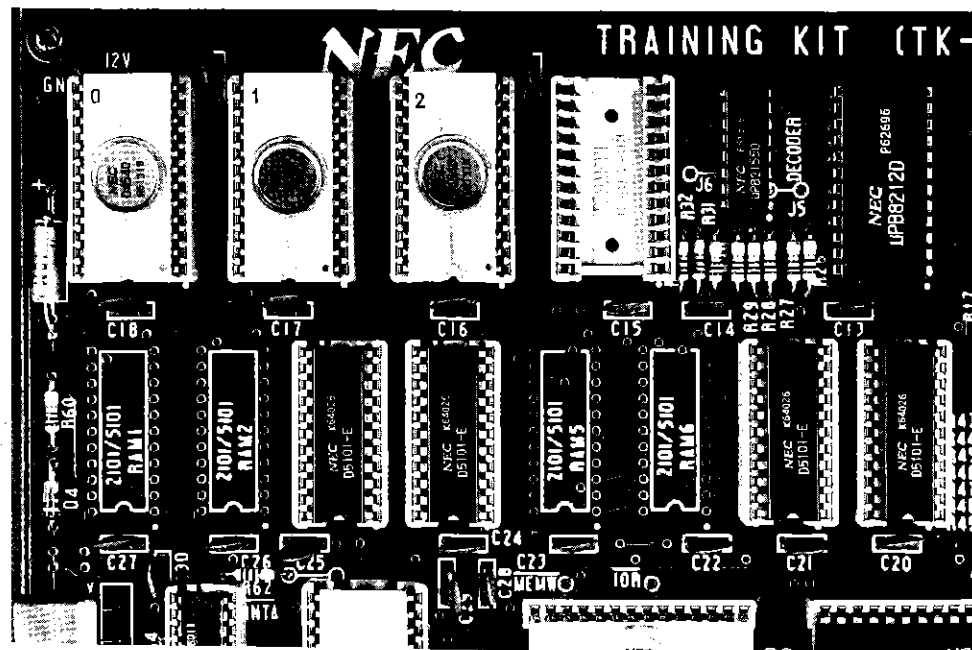
静電気に対して万全を期したい方は，台所に行ってください．多分そこにはステンレスの流し台があると思います．

（MOS ICの入力ゲートには過電圧に対する保護回路が入っていますので，一般的な注意事項を守っていれば普通は大丈夫ですが，上記の注意事項はいつも憶えておいてください．）

上記の点を考慮しながら，指定されたソケットへそれぞれのICを挿入してください．

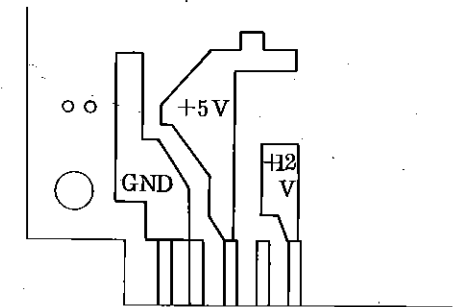
ICの方向はプリント基板に表示されていた通りの方向です．ICを挿入する場合には，少し差し込んだ状態で全てのピンが無理なく挿入されつつあることを確認してから押し込んでください．

PROM（454）3個には品名とは別に，0，1，2のマークが付けてありますので，それぞれPROM1，PROM2，PROM3の位置に取り付けてください．このPROMには基本的な動作を制御するモニタ・プログラムが分割して入っているため，正常な位置でないと動作しませんので注意してください．

写真2－3　PROM，RAMの実装

2.8　電源の取り付け

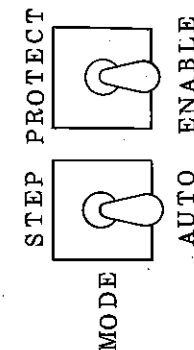
電源は外部から供給します．（1.5　電源に関する注意事項を参照して下さい）．プリント基板のGND，+5V，+12V用のハンダ付けエリアから直接リード線を引き出すか，または付属のプリント基板用コネクタの該当する端子を利用して供給します．

端子配列は付図Iを参照して下さい．

必要な電源		
+5V	±5％	0.9A以上
+12V	±5％	150mA以上

重要　電源は+5V→+12Vの順に投入してください（同時であれば可）． 電源切断は逆に+12V→+5Vの順に行ってください（同時であれば可）．

2.9　動作の確認

トグル・スイッチ２個はそれぞれENABLE，AUTO側に倒しておきます．電源投入後，RESETキーを押して離せば，８桁のLEDは全桁ゼロを表示します．LEDが点灯しないか，でたらめな値しか表示しない場合は，部品取り付けの再点検が必要です．

（スイッチの状態）

基本的な動作チェックを下に示す通りのキー操作で確認してください．LEDの表示の変化も示しておきます（××印は無視してください）．

キー入力操作　　　　LEDの表示

ここでMOD切換用トグル・スイッチをSTEP側に倒し，続けてキーを押していきます．

上記キー操作に従って，表示が正しく変化すれば，あなたの組み立てたキットはほぼ完全に動作しております．

2.10　トラブル対策

確認の結果，正常に動作していない場合には，次の点に注意してもう一度部品の取り付けに誤りがないかをチェックしてください．

(1)　ICは正しい位置に正しい方向で取り付けられているか
(2)　抵抗，コンデンサも正しく取り付けられているか
(3)　ハンダでショートしている箇所がないか

LED表示が全然なされない場合には，すぐに電源を切って電源系統に異常がないかを調べてくだ

さい．1桁だけしか表示していない場合は，ICタイマ（555）とカウンタ（223）の回りをよく調べてください．

全桁表示はするがキー入力を行ってもその数値が表示されない場合は，CPUが正常に動作していないことが考えられます．

この時トランジスタラジオを近づけますと，正常に動作していれば雑音が入り，キーを押した状態と離した状態では音色が異なるのがわかります．

図2－13　LSI，ICピン配列一覧表

μPD8080A

セントラル・プロセッサ・ユニット

ピン	信号	ピン	信号
1	A_{10}	40	A_{11}
2	(0V)V_{SS}	39	A_{14}
3	D_4	38	A_{13}
4	D_5	37	A_{12}
5	D_6	36	A_{15}
6	D_7	35	A_9
7	D_3	34	A_8
8	D_2	33	A_7
9	D_1	32	A_6
10	D_0	31	A_5
11	(−5V)V_{BB}	30	A_4
12	RESET	29	A_3
13	HOLD	28	VDD(+12V)
14	INT	27	A_2
15	ϕ_2	26	A_1
16	INTE	25	A_0
17	DBIN	24	WAIT
18	$\overline{WR}$	23	READY
19	SYNC	22	ϕ_1
20	(+5V)V_{CC}	21	HLDA

μPD8255

プログラマブル周辺インタフェース

ピン	信号	ピン	信号
1	PA_3	40	PA_4
2	PA_2	39	PA_5
3	PA_1	38	PA_6
4	PA_0	37	PA_7
5	$\overline{RD}$	36	$\overline{WR}$
6	$\overline{CS}$	35	RESET
7	GND	34	D_0
8	A_1	33	D_1
9	A_0	32	D_2
10	PC_7	31	D_3
11	PC_6	30	D_4
12	PC_5	29	D_5
13	PC_4	28	D_6
14	PC_0	27	D_7
15	PC_1	26	Vcc(+5V)
16	PC_2	25	PB_7
17	PC_3	24	PB_6
18	PB_0	23	PB_5
19	PB_1	22	PB_4
20	PB_2	21	PB_3

μPB8228

システムコントローラ・バスドライバ

ピン	信号	ピン	信号
1	$\overline{STSTB}$	28	Vcc(+5V)
2	HLDA	27	$\overline{I/OW}$
3	$\overline{WR}$	26	$\overline{MEMW}$
4	DBIN	25	$\overline{I/OR}$
5	DB_4	24	$\overline{MEMR}$
6	D_4	23	$\overline{INTA}$
7	DB_7	22	$\overline{BUSEN}$
8	D_7	21	D_6
9	DB_3	20	DB_6
10	D_3	19	D_5
11	DB_2	18	DB_5
12	D_2	17	D_1
13	DB_0	16	DB_1
14	GND	15	D_0

μPB8224

クロックジェネレータ・ドライバ

Pin	Signal	Pin	Signal
1	$\overline{RESET}$	16	Vcc(+5V)
2	$\overline{RESIN}$	15	XTAL1
3	RDYIN	14	XTAL2
4	READY	13	TANK
5	SYNC	12	OSC
6	ϕ_2(TTL)	11	ϕ_1
7	$\overline{STSTB}$	10	ϕ_2
8	GND	9	V_{DD}(+12V)

μPB8212

8ビットI/Oポート

Pin	Signal	Pin	Signal
1	$\overline{DS_1}$	24	Vcc
2	MD	23	$\overline{INT}$
3	DI_1	22	DI_8
4	DO_1	21	DO_8
5	DI_2	20	DI_7
6	DO_2	19	DO_7
7	DI_3	18	DI_6
8	DO_3	17	DO_6
9	DI_4	16	DI_5
10	DO_4	15	DO_5
11	STB	14	$\overline{CLR}$
12	GND	13	DS_2

μPD5101

1024ビット・スタティックRAM

Pin	Signal	Pin	Signal
1	A_3	22	Vcc(+5V)
2	A_2	21	A_4
3	A_1	20	R/W
4	A_0	19	$\overline{CE1}$
5	A_5	18	OD
6	A_6	17	CE2
7	A_7	16	DO_4
8	GND	15	DI_4
9	DI_1	14	DO_3
10	DO_1	13	DI_3
11	DI_2	12	DO_2

μPD454

2048ビット EE PROM

Pin	Signal	Pin	Signal
1	$\overline{CS}$	24	Vcc
2	A_0	23	D_0
3	A_1	22	D_1
4	A_2	21	D_2
5	A_3	20	D_3
6	A_4	19	D_4
7	A_5	18	D_5
8	A_6	17	D_6
9	A_7	16	D_7
10	P_G	15	V_{CG}
11	V_{CL}	14	V_{DD}
12	V_{BB}	13	Vss

μPB201C/D(7400)

14	13	12	11	10	9	8
Vcc	B_4	A_4	Y_4	B3	A3	Y3

1	2	3	4	5	6	7
A1	B1	Y1	A2	B_2	Y_2	GND

μPB202C/D(7410)

14	13	12	11	10	9	8
Vcc	A1	Y1	C3	B3	A3	Y3

1	2	3	4	5	6	7
B1	C1	A2	B2	C2	Y2	GND

μPB214C/D(7474)

14	13	12	11	10	9	8
Vcc	CLEAR2	D2	CLOCK2	PRESET2	Q2	$\overline{Q2}$

1	2	3	4	5	6	7
CREAR1	D1	CLOCK1	PRESET1	Q1	$\overline{Q1}$	GND

CLEAR D Q CLOCK $\overline{Q}$ PRESET / PRESET CLOCK Q D Q CLEAR

μPB215C/D(7401)

14	13	12	11	10	9	8
Vcc	Y_4	B_4	A_4	Y3	B3	A3

1	2	3	4	5	6	7
Y1	A1	B1	Y_2	A2	B_2	GND

μPB223C/D(7493A)

14	13	12	11	10	9	8
INPUT A	NC	A	D	GND	B	C

1	2	3	4	5	6	7
B INPUT	RO(1)	RO(2)	NC	Vcc	NC	NC

A B C D / INPUT A B / RORO (1)(2)

μPB238D(7438)

14	13	12	11	10	9	8
Vcc	B_4	A_4	Y_4	B3	A3	Y3

1	2	3	4	5	6	7
A1	B1	Y1	A2	B_2	Y_2	GND

μPB2155D(74155)

16	15	14	13	12	11	10	9
Vcc	DATA 2C	STROBE 2G	SELECT A	2Y3	2Y2	2Y1	2Y0

1	2	3	4	5	6	7	8
DATA 1C	STROBE 1G	SELECT B	1Y3	1Y2	1Y1	1Y0	GND

2C 2G A 2Y3 2Y2 2Y1 2Y0 / 1C 1G B 1Y3 1Y2 1Y1 1Y0

SN74LSO4

14	13	12	11	10	9	8
Vcc	6A	6Y	5A	5Y	4A	4Y

1	2	3	4	5	6	7
1A	1Y	2A	2Y	3A	3Y	GND

NE/SE555

Pin	Signal	Pin	Signal
1	Ground	8	Vcc
2	Trigger	7	Discharge
3	Out put	6	Threshold
4	Reset	5	Control Voltage

第3章　モニタプログラムとその操作方法

3.1　概　　要

どんなコンピュータでもプログラムなしには，何も仕事をすることができません．そのプログラムはメモリに書かれますので，コンピュータにはプログラムを"メモリに書く"とか，"メモリの内容を調べる"といった基本的な機能が必要となります．

さらに，プログラムがメモリに書き込まれても，そのプログラムが正しく，思った通りに動作してくれるかどうかチェックするための手段も必要です．このような機能は，複雑なハードウェアを備えれば実現できますが，ＴＫ－８０では，この基本的な処理の大部分をプログラムでソフトウェア的に実現しています．

このプログラムはモニタプログラムと呼ばれ，EEPROM（μPD454）に書き込まれた形で，キット部品の中に含まれております．（ソフトウェアが"部品"と同じようにＬＳＩの形で届けられるわけで，こういう所がマイクロコンピュータの便利な所です）．

3.2　基本的な操作方法

モニタの詳しい説明は後回しにして，早くこのコンピュータにプログラムを入力できるようにしたい方は，この項を読んでください．

(1)　電源の投入

電源は＋５Ｖを先に，＋１２Ｖを後で投入します．順序をつけにくいときは，＋５Ｖ,+12Vを同時に投入してください．

(2)　電源を同時に投入した場合

電源を同時に投入した場合には，自動リセット回路（パワー・オン・リセット）が働いて，コンピュータにリセットがかかりますが，電源投入後は，一応 RESET キーにより，コンピュータにリセットをかけてから操作に移るくせをつけましょう．

(3)　順序をつけて投入した場合

＋１２Ｖを遅らせて投入した場合には，自動リセットは働きませんので，必ず RESET キーを押してください．

(4)　モニタ・プログラム・スタート

リセットがかけられると，このコンピュータのモニタプログラムが動作をはじめ，ＬＥＤ表示部のすべての桁に"０"が表示されます．

それでは次に簡単なプログラミングの例を示しますので，指示通りにキー操作を行ってください．一通り操作方法がマスターできます．

3.3　基本的なプログラミング操作例

(1)　8200番地からプログラムを書いていきます（この通りの値をプログラムしていってください）.

[▨] [▨] [3] [E] [WRITE INCR]　←　間違ったデータは無視して，続けて入力してください．最後の2桁だけが有効です．

[B] [0] [WRITE INCR]

[3] [E] [WRITE INCR]

[C] [C] [WRITE INCR]

[C] [3] [WRITE INCR]

[0] [0] [WRITE INCR]

[8] [2] [WRITE INCR]

```
8200 MVI A,0AAH
     MVI A,0BBH
     MVI A,0CCH
     JMP 8200H
```

(2)　8200番地から書かれた内容をチェック

データ部の表示

[8] [2] [0] [0]

[ADRS SET]　　003E

[READ INCR]　　3EAA　　OK

[READ INCR]　　AA3E　　OK

[READ INCR]　　3Eb0　←　間違っていたので“BB”に直したい

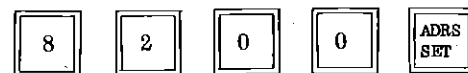
このときはアドレスはすでにセットされていますので，正しいデータ2桁と書き込みキーを押すだけで，データは変更されます．

OK

(3)　8200番地からこの命令を1ステップ（1命令ずつ）実行します．

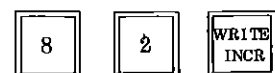
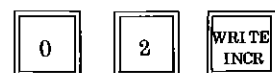
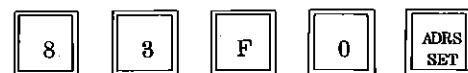
モードスイッチをSTEP側にしておきます．

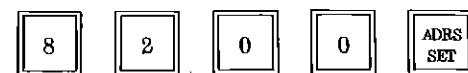
8200～8206番地を繰り返し実行し，データ部の表示はAA，BB，CCのように変化していきます．

(4) このプログラムをAUTOで実行させてみます．

モードスイッチをAUTO側にしておきます．

[RUN]

これでプログラムはループを繰り返し実行していますが，外から見ているだけでは，本当に実行しているかどうかわかりません．

(5) ループをN回実行するまではAUTOで，それ以後はステップ動作をさせてみます（8202番地を１００回通過したら，ステップ動作に入ることにします）．

ここでモードスイッチをＳＴＥＰにします．

[RUN]

プログラムはあっという間に１００回実行されて
アドレスは８２０４
データはＢＢ××
が表示されてストップします．
この先ステップ動作が可能です．

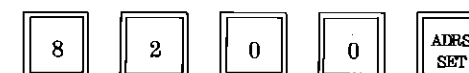
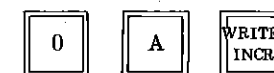
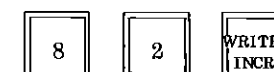
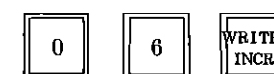
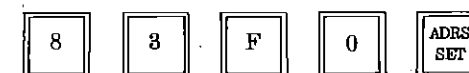
[RET]

[RET]

(6) ステップ動作でＣＰＵ内部のレジスタの動きをチェックします．

```
8200   MVI  B,0     0600
       MVI  C,0     0E00
       INR  B       04
       DCR  C       0D
       JMP  8204H   C30482
```

まず，このプログラムを８２００番地から書いてください．

今度は１０回繰り返し実行してから，ステップ動作に入るものとします．

モードスイッチをＳＴＥＰ側にしておきます．

[RUN] １０回実行されて止まります．この時のＢレジスタ，Ｃレジスタの値を調べてみます．

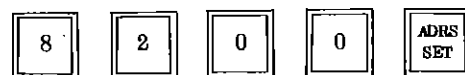
Ｂレジスタ，Ｃレジスタの内容は，それぞれ８３Ｅ９，８３Ｅ８番地に格納されていますので，この番地を読み出します．

データ表示部には，０ＡＦ６が表示されます．０ＡがＢレジスタの値，Ｆ６がＣレジスタの値です．Ｂ．Ｃレジスタだけでなく，ＣＰＵチップの中のすべてのレジスタはこのようにして調べることができます（詳しくは３．４．７レジスタの表示の項を参照してください）．

さらに続けて１ステップずつ実行させるには，[RET] キーを押します．１ステップ実行させるたびに，新しいレジスタの内容が前に調べた番地に更新されて書かれますので，もう一度同じようにアドレスをセットして，その番地を読み出せば，１ステップずつのレジスタの動きを調べることができます．

(7) 完成したプログラムをカセットテープにファイルします．

カセットテープとＴＫ－８０のインタフェースは第６章を参照してください．

８２００番地から８２３５番地までをひとまとめにしてファイルするものとします．

8	2	0	0	ADRS SET	スタートアドレスの設定
8	2	3	5		エンドアドレスの設定

ここでテープレコーダの録音を開始しますと，ピーという連続した音が書き込まれます．
VUメーターが適当な録音レベルを指示していることを確めてください．

[STORE DATA] キーを押すと，書き込みを開始します．

書き込み中はLED表示が消えるようになっており，終了しますと再び表示されます．

書き込みが不安な場合は，もう一度 [STORE DATA] キーを押せば，同じデータの書き込みが行われます．

(8) テープよりメモリへデータをロードします．

テープにファイルしたプログラムには，格納されるべき番地も書かれていますので，番地を指定する必要はありません．テープを再生状態にすると，ピーという連続音の入っている部分がデータより前に現われますので，[LOAD DATA] キーを押します．

データを読み取っている間は，LEDの表示は消えますが最後のデータであるサムチェック・データを読み取り，エラーがなければ，データのスタートアドレスとエンドアドレスがそれぞれアドレス表示部とデータ表示部に現われます．

エラーがあった場合には，

E．．．．．．

が表示されますので，もう一度読み込ませてください．

プログラムのスタート番地と読み込みデータのスタート番地が一致している場合には [RUN] キーを押せばすぐこのプログラムは走り出します．

読み込みエラーが続発するような場合には，テープへの録音レベルかテープの再生レベルが不適当な状態にあると思われますので，各レベルを調整した上でストア，ロードを行ってください．

これまでの操作ができれば，あなたはほぼ自由にプログラムを組んだり，デバグを行うことができます．

さらに深くモニタプログラムの機能を習得したい方は，第4章を学んでください．それよりも早くこのコンピュータに何かをやらせてみたい方は，別冊として用意されている．

“TK－80応用プログラム”を参考にして，実際にプログラムを入力してください．

3.4 プログラミングに関する基本的な注意事項

基本構成では，RAMは512バイトが実装されており，そのロケーションは，

8200～83FF （16進表示）

です．そのうちモニタプログラムがワーキングエリアとして，

83C7～83FF （16進表示）

番地を使用していますので，ユーザがプログラムを書けるロケーションは，

8200～83C6 （16進表示）

番地です．さらにユーザプログラムの中でスタックを使う命令（PUSH，CALL等）が実行されると，スタックが83C6番地から若い番地に向ってのびてきますので，スタックの使い方注(1)をマスターするまでは，できるだけ若い番地からプログラムを書く（つまり高い方の番地でスタック領域が多少ふえてもプログラムに影響しないようにしておきます）ようにしてください．

注(1) 4．2 サブルーチンの考え方の項を参照してください．

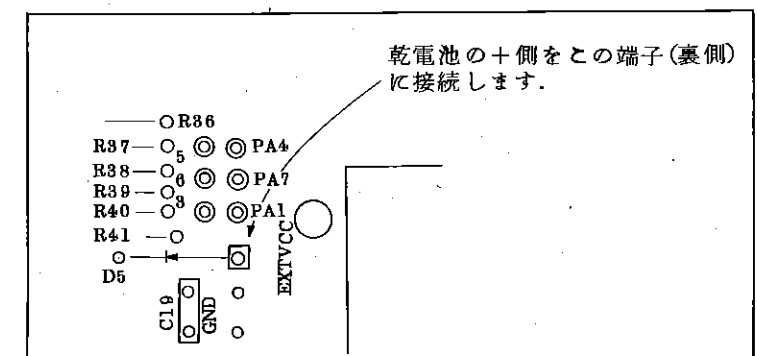
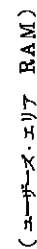
3.5 バッテリによるメモリデータの保存

一般の半導体RAMは電源が切れると，データも消えてしまいます．しかし，TK－80で使用しているCMOS RAM は，スタンバイ時（リード／ライト動作を行っていない状態）の消費電力が非常に少ないので，バッテリーで長時間データを保存させることができます．TK－80の場合には，乾電池2本（1.5V×2）を直列接続してEXT Vccの端子につないでおいてください．

そしてシステムの電源を切るときには，まず [RESET] キーを押し，RAM PROTECT／ENABLE スイッチをPROTECT側に倒してから，電源を切ってください．そのまま乾電池が消耗してしまわないかぎり，データは保存されます．

電源を再び投入する場合には，先に電源を投入し，[RESET] キーを押しながらスイッチをENABLE側に切り変えれば，再びRAMは元のデータをもったまま，使用可能な状態になります．

図3－1　モニタプログラムによる処理

図3－2　モニタプログラムのフローチャート

3.6　モニタプログラムの詳細な説明

これまで説明した操作は，モニタプログラムの内部的な処理には一切触れませんでした．

ここでは，さらに高度な使い方をなさりたい方，もしくはモニタプログラムの中身を勉強されたい方のために，モニタプログラムを少し詳しく説明しておきます．

3.6.1　モニタプログラムのスタート

モニタは，RESETキーが押された時，あるいはCPUプログラムカウンタが"0"になるような命令（JMP　0，RST　0，CALL　0等）が，実行された時にスタートします．

3.6.2　モニタプログラム・スタート時の初期値設定

モニタのエントリーは次の二つがあります．

(1)　0番地スタート

モニタを0番地より，スタートさせるとモニタは，そのワーキングエリアの注(1)データレジスタ，注(2)アドレスレジスタ，注(3)ブレーク・アドレス・レジスタ，注(4)ブレークカウンタ，注(5)キーフラグ，注(6)ディスプレイ・レジスタを零クリアします．

次に，注(7)スタックポインタのセーブエリアに，最初のユーザスタックとなる番地"83C7"を書き込みその後，スタックポインタを，モニタ専用エリア"83D1"にセットします．

以上のイニシャライズが終了すると，零クリアされているディスプレイレジスタの内容を，LEDディスプレイに表示し，キー入力待ちの状態になります．

注(1)　データキーより入力されたデータおよびメモリよりリードされたデータが，セットされるソフトウェア上のレジスタ

(2)　モニタがメモリに対して処理を行う時に参照するレジスタで［ADRS SET］キーを押すことによって，データレジスタのデータをセットすることができます．

(3)　ブレーク番地をセットするレジスタ

(4)　ブレーク動作時に，ループ回数をセットするレジスタ

(5)　モニタがキーボードをセンスする時に参照されるフラグ

(6)　LEDディスプレイに表示するためのデータをセットするレジスタ

(7)　ステップおよびブレーク時にスタックポインタを退避させるためにとられているエリア

(2)　8番地スタート

モニタを8番地よりスタートさせると，上記のイニシャライズは行わず，スタックポインタを，モニタ専用エリア"83D1"にセットします．

次に，その時ディスプレイレジスタにセットされているデータを，LEDディスプレイに表示し，キー入力待ちの状態になります．

3.6.3　データのセット

モニタは，入力データ用に2ワードのデータレジスタをもっています．このレジスタは，4ビット構成でキーより入力された16進数データを4桁まで，記憶しておくことができます．

データキー（［0］～［F］）が，押されるとデータレジスタは，1桁上位にシフトされ，入力されたデータはデータレジスタの最下位にセットされます．

キー入力	ADDRESS	DATA
［RESET］	0000	0000
［1］	0000	0001
［2］	0000	0012

3.6.4　キーコマンド

［ADRS SET］　アドレス　セット

［READ INCR］　アドレス・インクリメント＆メモリリード

［READ DECR］　アドレス・ディクリメント＆メモリリード

［WRITE INCR］　メモリライト＆アドレス・インクリメント

［RUN］　RUN

［RET］　RETURN

［LOAD DATA］　ロード　データ

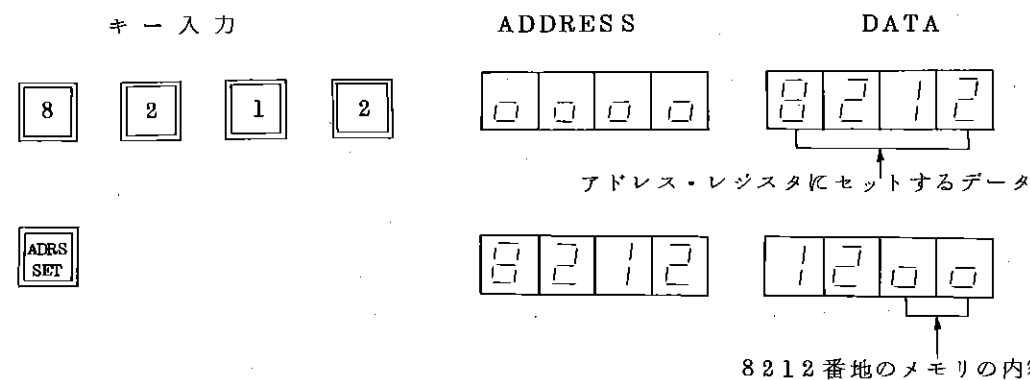
［STORE DATA］　ストア　データ

(1)　アドレスセット

データレジスタにセットされたデータをアドレスレジスタにセットし，その番地のメモリの

内容をリードします．

キー入力	ADDRESS	DATA
8 2 1 2	0000	8212
ADRS SET	8212	1200

アドレス・レジスタにセットするデータ

8212番地のメモリの内容

データレジスタに，これから処理を行おうとするアドレスを，16進数4桁でセットし，ADRS SET キーを押すと，データレジスタにセットされたアドレスデータが，アドレスレジスタにセットされ，その番地のメモリの内容がリードされます．

この時データレジスタは，2桁上位にシフトされリードされたデータは，データレジスタの下位2桁にセットされます．

以上の処理が終了すると，データレジスタとアドレスレジスタの内容は，LEDディスプレイに表示されます．

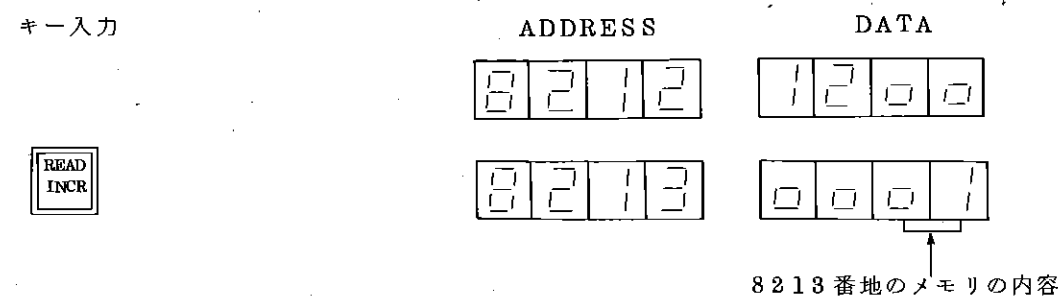
(2) アドレス・インクリメント&メモリリード

アドレスレジスタの内容をインクリメントし，その番地のメモリの内容をリードします．

READ INCR キーが押されると，アドレスレジスタの内容をインクリメントし，さらに更新された番地のメモリの内容がリードされます．

この時データレジスタは，2桁上位にシフトされリードされたデータは，データレジスタの下位2桁にセットされます．

以上の処理が終了すると，データレジスタとアドレスレジスタの内容は，LEDディスプレイに表示されます．

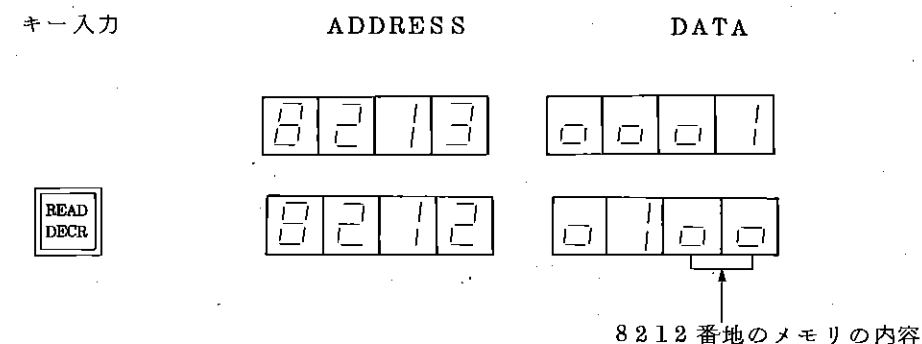
キー入力	ADDRESS	DATA
	8212	1200
READ INCR	8213	0001

8213番地のメモリの内容

(3) アドレス・ディクリメント&メモリリード

アドレスレジスタの内容をディクリメントし，その番地のメモリの内容をリードします．

READ DECR キーが押されると，アドレスレジスタの内容をディクリメントし，さらに更新された番地のメモリの内容がリードされます．

この時データレジスタは，2桁上位にシフトされリードされたデータは，データレジスタの下位2桁にセットされます．

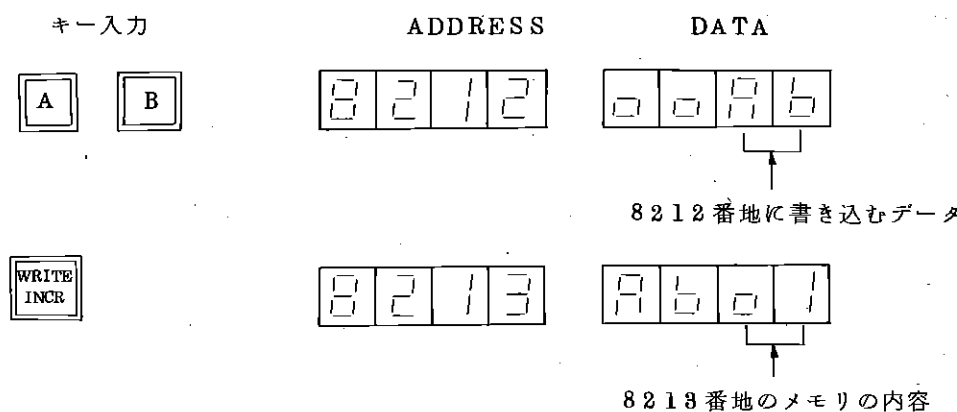
以上の処理が終了すると，データレジスタとアドレスレジスタの内容は，LEDディスプレイに表示されます．

(4) メモリライト&アドレス・インクリメント

アドレスレジスタの内容によって指定された番地のメモリに，データレジスタの下位2桁にセットされたデータをライトします．

データレジスタの下位2桁に，ライトするデータをセットして WRITE INCR キーが押されると，アドレスレジスタによって，指定された番地のメモリにデータレジスタの下位2桁に，セットされているデータが書き込まれます．

次にアドレスレジスタの内容は，インクリメントされ更新された番地のメモリの内容が，リードされます．

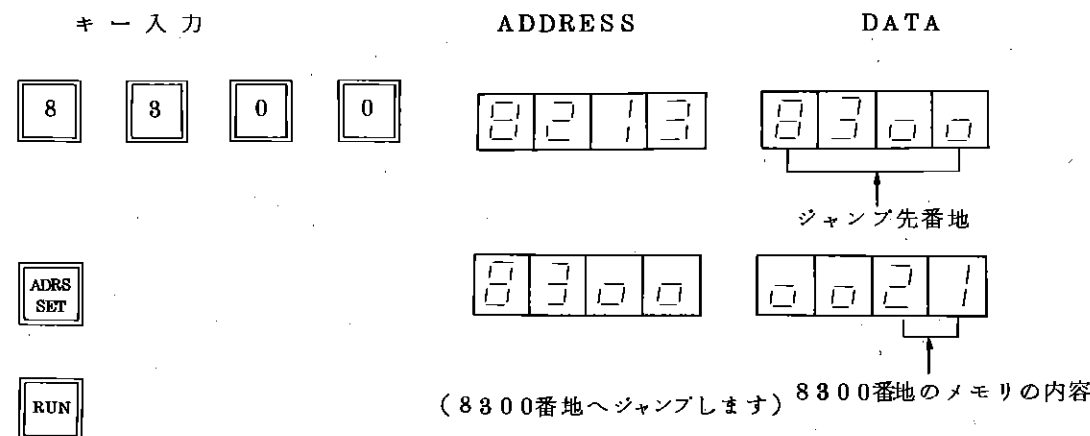
この時データレジスタは，2桁上位にシフトされ，リードされたデータは，データレジスタの下位2桁にセットされます．

以上の処理が終了すると，データレジスタとアドレスレジスタの内容は，LEDディスプレイに表示されます．

(5) RUN

アドレスレジスタの内容によって指定された番地へジャンプします．

データレジスタに，ジャンプ先の番地を16進4桁でセットして，[ADRS SET]キーによりアドレスレジスタにジャンプ先の番地をセットします．

この後[RUN]キーを押すと，レジスタセーブエリアに退避されているレジスタの内容をCPUレジスタに復帰して，アドレスレジスタにセットされている番地にジャンプします．

なおジャンプする直前に，EI命令を実行するため割り込みイネーブルの状態でジャンプしていきます．

(6) RETURN

[RET]キーが押されると，レジスタ・セーブ・エリアに退避されているレジスタの内容をCPUレジスタに復帰して，退避されていたプログラムカウンタによって指定される番地にジャンプします．

なおジャンプする直前に，EI命令を実行するため，割り込みイネーブルの状態でジャンプしていきます．

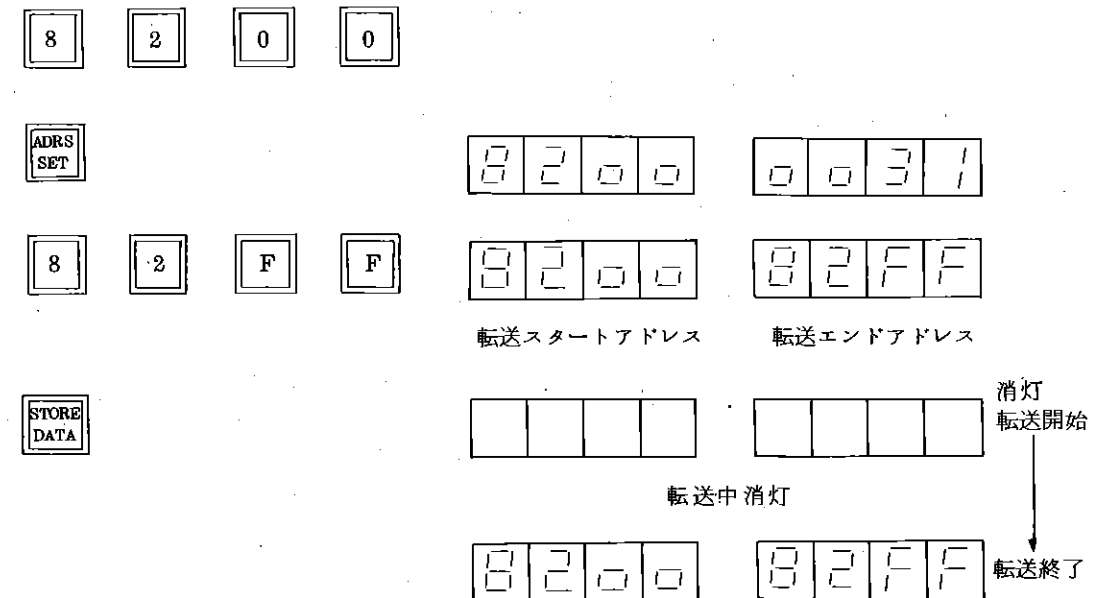
(7) ストア・データ

アドレスレジスタで指定された番地からデータレジスタで指定された番地までのメモリの内容を，シリアル信号に変換してPPI（μPD8255）のポートC（PC0）に出力します．

この信号をカセット・インタフェースにより（第6章を参照して下さい），オーディオ帯域の信号に変換してカセットテープに録音します．

データレジスタにこれから転送しようとするデータの格納されている先頭番地をセットし，[ADRS SET]キーを押し，転送スタートアドレスをアドレスレジスタにセットします．この後データレジスタに転送しようとするデータの格納されている最終番地をセットします．

ここで[STORE DATA]キーを押すと，LEDディスプレイの表示が消えデータ転送がはじまります．

データ転送が終了すると，LEDディスプレイの表示が再び点灯し，キー入力待ちの状態になります．

(8) ロード・データ

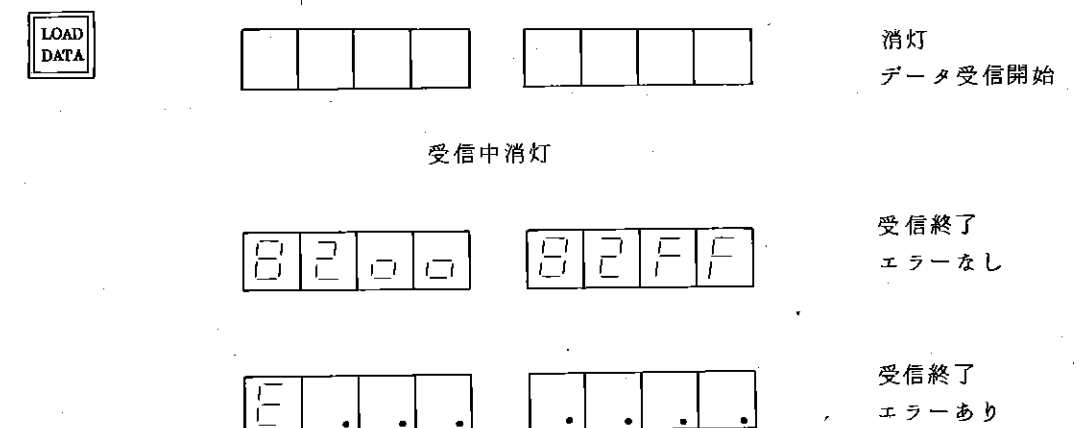
カセットテープからのデータを，データ中で指定された番地のメモリへロードします．

カセットテープをスタートさせ，発振音を確認した後に[LOAD DATA]キーを押すと，LED表示が消え，カセットテープよりロード先のアドレスデータを受信し，以降のデータをメモリにロードします．

ロードが終了すると，受信にエラーがあったかどうかをチェックし，エラーがない場合は，ロードされたデータの先頭アドレスをアドレスレジスタに，最高番地をデータレジスタにセットし，アドレスレジスタ，データレジスタの内容をLEDディスプレイに表示します．

この状態で[RUN]キーを押すことにより，今ロードしたプログラムを実行させることができます．(ただし，ロード開始番地がプログラムのスタート番地に一致している場合)．

受信にエラーがあった場合は，LEDディスプレイにエラー・メッセージを表示します．

キー入力	ADDRESS	DATA	
LOAD DATA			消灯 データ受信開始
	受信中消灯		
	8200	82FF	受信終了 エラーなし
	E . . .	. . . .	受信終了 エラーあり

3.6.5 ステップ動作

ＴＫ－８０モニタは，割り込みによりプログラムをステップさせることができます．

ステップ動作を行わせる場合には，モニタのワーキングエリアにあるブレークカウンタ(83F2番地)を零クリアするか，又は[RESET]キーを押し(モニタプログラムが０番地よりスタートすると必ず零クリアされます．)さらにモードスイッチを“ STEP”にします．

この状態でアドレスレジスタに，プログラムのスタート番地をセットして[RUN]キーを押すとプログラムをステップして(１インストラクション実行して)，モニタに戻ってきます．この時CPUのすべてのレジスタは，レジスタ・セーブ・エリアに退避されます．

さらにアドレスレジスタには，その時のプログラムカウンタの内容が，データレジスタの上位２桁にはアキュムレータの内容が，データレジスタの下位２桁にはフラグレジスタの内容がセットされ，ＬＥＤディスプレイに表示されます．

この時モニタのキーコマンドにより，各部の動作(メモリ，レジスタの内容等)を確認することができます．

この後[RET]キーを押すと，退避されていたレジスタの内容をすべてＣＰＵに復帰して，次のインストラクションを実行して，再びモニタに戻ってきます．

この場合，リターン先の番地をセットする必要はありません

このようにして，プログラムを次々とステップさせていくことができます．

3.6.6 ブレーク動作

ＴＫ－８０モニタは，ブレークポインタを１つもっており，ここにセットされた番地において，ブレークさせることができます．またブレークポインタとともに，ブレークカウンタをもっているためブレークのループ回数を設定することができます．

ブレークポインタは，モニタのワーキングエリア内の８３Ｆ０番地と８３Ｆ１番地に置かれており，ここにブレークさせる番地を書き込みます．

またブレークカウンタは８３Ｆ２番地に置かれており，ここにループ回数をセットします．

(1) ブレークポインタおよびブレークカウンタのセット

データレジスタに８３Ｆ０をセットして[ADRS SET]キーを押し，アドレスレジスタにブレークポインタの番地をセットします．

データレジスタの下位２桁に，ブレークアドレスの下位２桁をセットして[WRITE INCR]キーを押し，続いてデータレジスタの下位２桁に，ブレークアドレスの上位２桁をセットして[WRITE INCR]キーを押して，ブレークポインタにブレークアドレスを書き込みます．

データレジスタの下位２桁に，ループ回数を１６進数でセットして[WRITE INCR]キーを押して，ブレークカウンタにループ回数を書き込みます．

キー入力	ADDRESS	DATA
8 3 F 0	0000	83F0 (ブレークポインタの番地)
ADRS SET	83F0	F000
0 3	83F0	0003
WRITE INCR	83F1	0300
8 3	83F1	0083
WRITE INCR	83F2	8300
0 3	83F2	0003
WRITE INCR		

(2) 動　作

前述(1)に従って，ブレークポインタおよびブレークカウンタをセットした後，アドレスレジスタにプログラムのスタート番地をセットし，モードスイッチを“ＳＴＥＰ”にして[RUN]キーを押すと，ブレークアドレスに対応するインストラクションを，ループ回数だけ実行した直後にブレークして，モニタに戻ってきます．

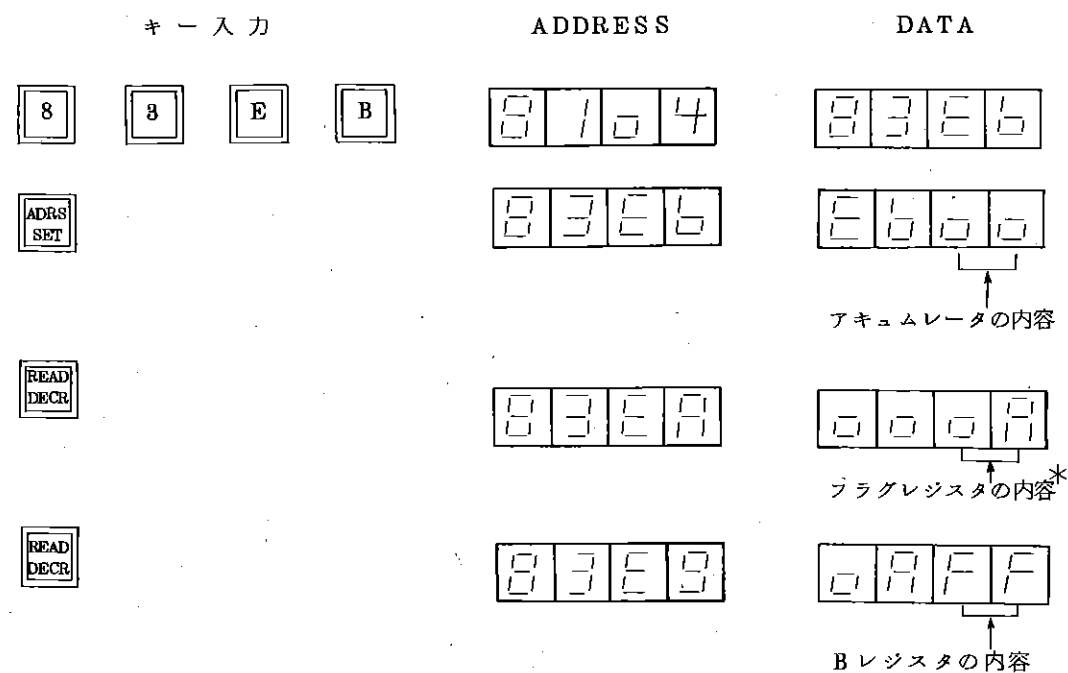
注　ブレークアドレスは，必ず各インストラクションのオペレーションコードの格納されている番地でなければなりません．

3.6.7 レジスタの表示

ステップ動作およびブレーク動作を行ってモニタに戻った時，すべてのＣＰＵレジスタはモニタ・ワーキング・エリア内のレジスタ・セーブ・エリアに退避されます．

この時各レジスタは，次の番地に退避されます．

８３ＥＢ	番地	アキュムレータ
８３ＥＡ	番地	フラグレジスタ＊
８３Ｅ９	番地	Ｂ　レジスタ
８３Ｅ８	番地	Ｃ　レジスタ
８３Ｅ７	番地	Ｄ　レジスタ
８３Ｅ６	番地	Ｅ　レジスタ
８３Ｅ５	番地	Ｈ　レジスタ
８３Ｅ４	番地	Ｌ　レジスタ
８３Ｅ３	番地	スタックポインタ〔上位〕
８３Ｅ２	番地	スタックポインタ〔下位〕
８３Ｅ１	番地	プログラムカウンタ〔上位〕
８３Ｅ０	番地	プログラムカウンタ〔下位〕

退避されているレジスタの内容は，各レジスタに相当するメモリの内容をモニタのキーコマンドにより，ＬＥＤディスプレイに表示させることができます．

＊　各フラグはフラグレジスタのビットと次のように対応します．

F_7	F_6	F_5	F_4	F_3	F_2	F_1	F_0
S	Z	SUB	CY_4	“1”	P	“1”	C

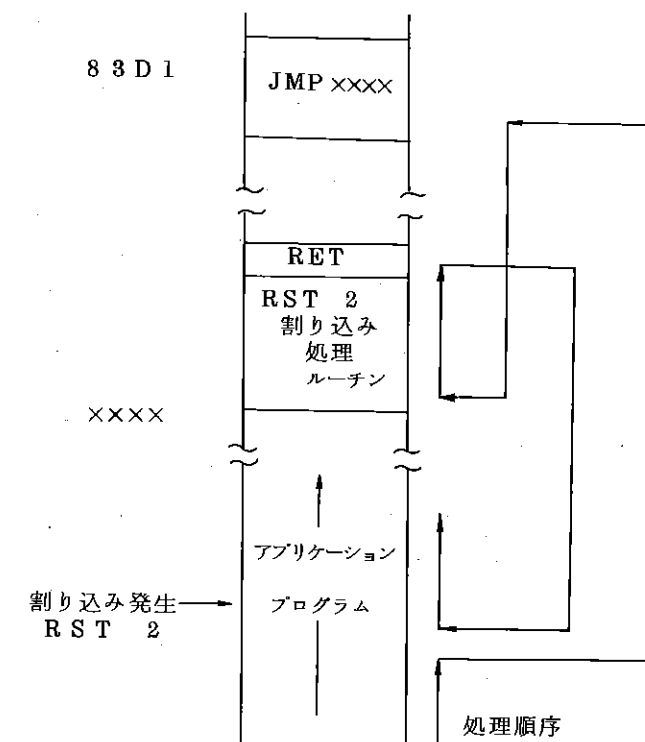
また，レジスタ・セーブ・エリア内にデータを書き込む（書き換える）と，RUN キーあるいは RET キーが押された時，ＣＰＵレジスタにはレジスタセーブエリア内に新しく書き込まれたデータを復帰させて，ジャンプしていきます．

つまりプログラムにジャンプする前に，モニタによってＣＰＵレジスタをイニシャライズすることができることになります．

3.6.8 リスタート・ジャンプ・テーブル

ＴＫ－８０モニタにおいて，８種類あるリスタート命令のうち５種類を開放しています．

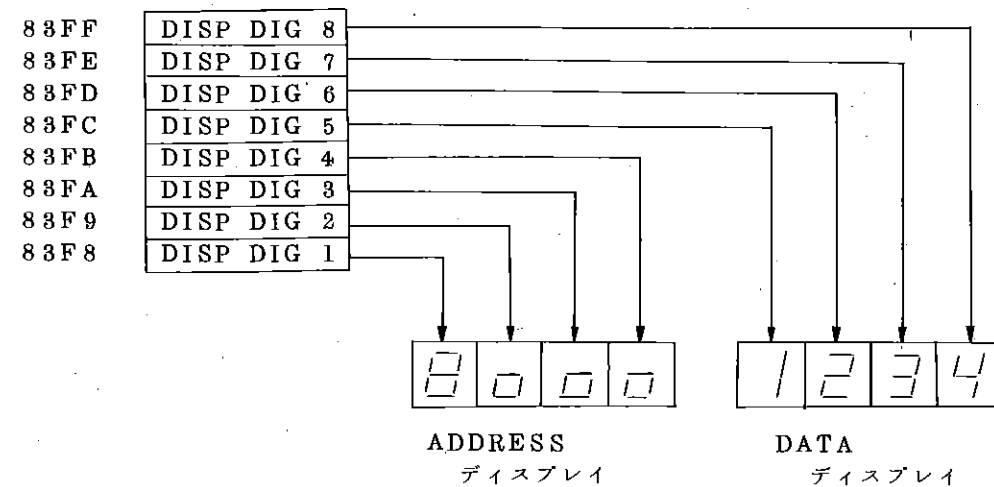
これらのリスタート命令を実行すると，おのおの次に示す番地に無条件ジャンプしてきます．従ってこのエリアに各処理ルーチンへのジャンプ命令を書き込んでおくことにより，各処理を実行することができます．

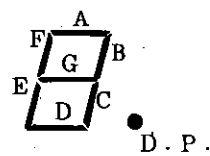
ＲＳＴ　２	８３Ｄ１	番地
ＲＳＴ　３	８３Ｄ４	番地
ＲＳＴ　４	８３Ｄ７	番地
ＲＳＴ　５	８３ＤＡ	番地
ＲＳＴ　６	８３ＤＤ	番地

3.6.9 LEDディスプレイへのデータの表示

TK−80は，モニタ・ワーキング・エリア内のセグメント・データ・バッファ（83F8番地〜83FF番地）の内容を，DMA転送によって常時表示しています．

表示は7セグメントのLED表示素子を使用し，ダイナミック点灯させています．

実際にデータをLEDディスプレイに表示させるためには，表示させるデータを後述の表示用データ[注(1)]（セグメントデータ）に変換して，上記のエリアに転送するだけでよく，表示のための特別なプログラムを書く必要はありません．

セグメントデータバッファは，次のようにLEDディスプレイの各桁に対応します．

注(1) セグメントデータ

1つの文字は，1ワード（8ビット）のデータで表示されますが，このデータは各ビットが次のように各セグメントに対応して構成されます．

A	:	BIT0
B	:	BIT1
C	:	BIT2
D	:	BIT3
E	:	BIT4
F	:	BIT5
G	:	BIT6
D.P.	:	BIT7

セグメントデータは，点灯させるセグメントに対応するビットを"1"とし，点灯させないセグメントに対応するビットを"0"として構成します．

例えば，"0"という文字に対応するセグメントデータは次のように構成されます．

A	:	（消灯）	BIT0	0	C
B	:	（消灯）	BIT1	0	
C	:	（点灯）	BIT2	1	
D	:	（点灯）	BIT3	1	

E	:	（点灯）	BIT4	1	5
F	:	（消灯）	BIT5	0	
G	:	（点灯）	BIT6	1	
D.P.	:	（消灯）	BIT7	0	

文字	セグメントデータ	文字	セグメントデータ
0	5C	8	7F
1	06	9	6F
2	5B	A	77
3	4F	b	7C
4	66	C	39
5	6D	d	5E
6	7D	E	79
7	27	F	71

3.7 TK−80メモリマップ

(1) RAMメモリマップ

アドレス	容量（バイト）	RAM &ROM	備考
FFFF ↑ 8400	31K	——	ブランク
83FF ↑ 83E0	32	RAM	モニタ・ワーキング・エリア
83DF ↑ 83D1	15	RAM	RST ジャンプテーブル
83D0 ↑ 83C7	10	RAM	モニタ・スタック・エリア
83C6 ↑ 8000	967	RAM	ユーザーズ・エリア
7FFF ↑ 0400	31K	—	ブランク
03FF ↑ 0300	256	EEPROM	ユーザーズ・エリア
02FF ↑ 0000	768	EEPROM	モニタ

(2) モニタ・ワーキング・エリア・メモリ・マップ

アドレス	シンボル	備考
83FF	DISP DIG 8	セグメント・データ・バッファ
FE	DISP DIG 7	
FD	DISP DIG 6	
FC	DISP DIG 5	
FB	DISP DIG 4	
FA	DISP DIG 3	
F9	DISP DIG 2	
83F8	DISP DIG 1	
83F7	DISP WORD 4	ディスプレイ レジスタ
F6	DISP WORD 3	
F5	DISP WORD 2	
83F4	DISP WORD 1	ディスプレイレジスタ
83F3	KEY FLAG	キーインプット・フラグ
83F2	BRKCT	ブレーク・カウンタ
83F1	BRKAD 〔HI〕	ブレーク・アドレス・レジスタ 上位
83F0	BRKAD 〔LO〕	下位
83EF	ADRES 〔HI〕	アドレスレジスタ 上位
83EE	ADRES 〔LO〕	下位
83ED	DATA 〔HI〕	データレジスタ 上位
83EC	DATA 〔LO〕	下位
83EB	A	CPU レジスタ・セーブ・エリア
EA	F	
E9	B	
E8	C	
E7	D	
E6	E	
E5	H	
E4	L	
E3	SP 〔HI〕	
E2	SP 〔LO〕	
E1	PC 〔HI〕	
83E0	PC 〔LO〕	
DF	RST 6	RST 6 ジャンプテーブル
DE		
83DD		
DC	RST 5	RST 5 ジャンプテーブル
DB		
83DA		

アドレス	シンボル	備考
D9 D8 83D7	RST 4	RST 4 ジャンプテーブル
D6 D5 83D4	RST 3	RST 3 ジャンプテーブル
D3 D2 83D1	RST 2	RST 2 ジャンプテーブル
83D0 CF CE CD CC CB CA C9 C8 83C7	MONSP	モニタ スタックエリア
83C6 ≈ 8000	USESP ≈	↓ユーザーズ スタック ≈ ユーザーズエリア

3.8 モニタ・アセンブル・リスト

```
*****   UCOM - 8    ASSEMBL    LIST   *****                    P 0001

0001                   ::::::::::::::::::::::::::::::::::::::::::::::::
0002                   ::              MONITER  VER 1.0              ::
0003                   ::                  1976-7                    ::
0004                   ::                          TYPE B            ::
0005                   ::::::::::::::::::::::::::::::::::::::::::::::::
0006                          ORG   0
0007  0000   3E92             MVI   A,92H       ; CONTROL WORD FOR 8255
0008  0002   D3FB             OUT   OFBH        ; PROGRAM TO 8255
0009  0004   C3               JMP   MONST
             3B00
0010                          ORG   8H
0011  0008   3E92             MVI   A,92H       ; CONTROL WORD FOR 8255
0012  000A   D3FB             OUT   OFBH        ; PROGRAM TO 8255
0013  000C   C3               JMP   START
             5100
0014                          ORG   10H
0015  0010   C3               JMP   RST2
             D183
0016                          ORG   18H
0017  0018   C3               JMP   RST3
             D483
0018                          ORG   20H
0019  0020   C3               JMP   RST4
             D783
0020                          ORG   28H
0021  0028   C3               JMP   RST5
             DA83
0022                          ORG   30H
0023  0030   C3               JMP   RST6
             DD83
0024                          ORG   38H
0025  0038   C3               JMP   BRENT
             5101
0026                   ::
0027                   ::  INITIALIZE ROUTIN
0028                   ::
0029  003B   3EFF      MONST: MVI   A,OFFH
0030  003D   D3FA             OUT   OFAH        ; PORT C,BIT 0  INITIALIZE
0031  003F   21               LXI   H,DATA
             EC83
0032  0042   060C             MVI   B,12
0033  0044   AF               XRA   A
0034  0045   77               MOV   M,A
0035  0046   23               INX   H
0036  0047   05               DCR   B
0037  0048   C2               JNZ   $-3
             4500
0038  004B   21               LXI   H,USESP
             C783
0039  004E   22               SHLD  SSAVE       ; SET UP FOR USER STACK
             E283
```

```
0040                     ::
0041                     :: MONITOR START
0042                     ::
0043  0051   3EFF        START: MVI  A,0FFH
0044  0053   D3FA               OUT  0FAH        ; PORT C,BIT 0  INITIALIZE
0045  0055   31                 LXI  SP,MONSP    ; SP INITIALIZE
             D183
0046  0058   CD                 CALL SEGCG       ; SEGMENT CONVERT
             C001
0047  005B   CD                 CALL KEYIN       ; KEY INPUT
             1602
0048  005E   47                 MOV  B,A
0049  005F   E610               ANI  10H
0050  0061   CA                 JZ   DIGIT       ; IF ZERO INPUT DATA=0--->F
             8400
0051  0064   78                 MOV  A,B
0052  0065   E60F               ANI  0FH
0053  0067   0600               MVI  B,0         ; B=0
0054  0069   87                 ADD  A
0055  006A   4F                 MOV  C,A
0056  006B   21                 LXI  H,TABL
             7400
0057  006E   09                 DAD  B
0058  006F   7E                 MOV  A,M
0059  0070   23                 INX  H
0060  0071   66                 MOV  H,M
0061  0072   6F                 MOV  L,A
0062  0073   E9                 PCHL
0063  0074   CC00        TABL:  DW   GOTO
0064  0076   F901               DW   RESRG
0065  0078   9400               DW   ADSET
0066  007A   B800               DW   ADDCX
0067  007C   9D00               DW   ADINX
0068  007E   C200               DW   MEMW
0069  0080   D500               DW   STAPE
0070  0082   0701               DW   LTAPE
0071  0084   CD          DIGIT: CALL SHIFT       ; DATA REG SHIFT (4 BITS)
             B501
0072  0087   3A                 LDA  DATA
             EC83
0073  008A   B0                 ORA  B
0074  008B   32                 STA  DATA        ; INPUT DATA SET
             EC83
0075  008E   CD                 CALL RGDSP       ; ADDRESS & DATA REG DISPLAY
             A101
0076  0091   C3                 JMP  START
             5100
0077                     ::
0078                     :: ADDRESS SET
0079                     ::
0080  0094   2A          ADSET: LHLD DATA        ; HL=DATA REG
```



```
             EC83
0081  0097   22                 SHLD ADRES       ; STORE HL TO ADDRESS REG
             EE83
0082  009A   C3                 JMP  ADINX+4     ; MEMORY READ & ADDRESS DISPLAY
             A100
0083                     ::
0084                     :: MEMORY READ & ADDRESS INCREMENT
0085                     ::
0086  009D   2A          ADINX: LHLD ADRES       ; HL=ADDRESS REG
             EE83
0087  00A0   23                 INX  H           ; ADDRESS INCREMENT
0088  00A1   CD                 CALL MEMR        ; MEMORY READ
             AD00
0089  00A4   22          ADSTR: SHLD ADRES       ; STORE HL TO ADDRESS REG
             EE83
0090  00A7   CD                 CALL RGDSP       ; ADDRESS & DATA DISPLAY
             A101
0091  00AA   C3                 JMP  START
             5100
0092  00AD   3A          MEMR:  LDA  DATA
             EC83
0093  00B0   32                 STA  DATA+1      ; DATA REG SHIFT
             ED83
0094  00B3   7E                 MOV  A,M         ; MEMORY READ
0095  00B4   32                 STA  DATA        ; DATA--->DATA REG
             EC83
0096  00B7   C9                 RET
0097                     ::
0098                     :: MEMORY READ & ADDRESS DECREMENT
0099                     ::
0100  00B8   2A          ADDCX: LHLD ADRES       ; HL=ADDRESS REG
             EE83
0101  00BB   2B                 DCX  H           ; ADDRESS DECREMENT
0102  00BC   CD                 CALL MEMR        ; MEMORY READ
             AD00
0103  00BF   C3                 JMP  ADSTR
             A400
0104                     ::
0105                     :: MEMORY WRITE
0106                     ::
0107  00C2   2A          MEMW:  LHLD ADRES       ; HL=ADDRESS REG
             EE83
0108  00C5   3A                 LDA  DATA        ; A=DATA REG
             EC83
0109  00C8   77                 MOV  M,A         ; DATA WRITE
0110  00C9   C3                 JMP  ADINX
             9D00
0111                     ::
0112                     :: MONITOR TO USER CONTROL ROUTIN
0113                     ::
0114  00CC   2A          GOTO:  LHLD ADRES       ; HL=ADDRESS REG
```

```
              EE83
0115  00CF    22            SHLD PSAVE     ; (HL)--->(PC) SAVE AREA
              E083
0116  00D2    C3            JMP  RESRG     ; REGISTER RESTORE
              F901
0117                 ;                     & GO TO USER ROUTINE
0118                 ;;
0119                 ;; STORE DATA TO TAPE
0120                 ;;
0121  00D5    0E00   STAPE: MVI  C,0       ; C=CHECKSUM REGISTER
0122  00D7    2A            LHLD DATA      ; HL=END ADDRESS
              EC83
0123  00DA    EB            XCHG           ; DE=END ADDRESS
0124  00DB    2A            LHLD ADRES     ; HL=START ADDRESS
              EE83
0125  00DE    7C            MOV  A,H
0126  00DF    CD            CALL CKSMO     ; START ADDRESS [HI] OUT
              4101
0127  00E2    7D            MOV  A,L
0128  00E3    CD            CALL CKSMO     ; START ADDRESS [LO] OUT
              4101
0129  00E6    7A            MOV  A,D
0130  00E7    CD            CALL CKSMO     ; END ADDRESS [HI] OUT
              4101
0131  00EA    7B            MOV  A,E
0132  00EB    CD            CALL CKSMO     ; END ADDRESS [LO] OUT
              4101
0133  00EE    2B            DCX  H
0134  00EF    23     TAPE1: INX  H
0135  00F0    7E            MOV  A,M
0136  00F1    CD            CALL CKSMO     ; CHECKSUM & DATA OUT
              4101
0137  00F4    7D            MOV  A,L
0138  00F5    BB            CMP  E         ; START[LO],END[LO] COMPARE
0139  00F6    C2            JNZ  TAPE1
              EF00
0140  00F9    7C            MOV  A,H
0141  00FA    BA            CMP  D         ; START[HI],END[HI] COMPARE
0142  00FB    C2            JNZ  TAPE1
              EF00
0143  00FE    79            MOV  A,C
0144  00FF    2F            CMA
0145  0100    3C            INR  A
0146  0101    CD            CALL CKSMO     ; CHECKSUM OUT
              4101
0147  0104    C3            JMP  START     ; END STORE TAPE
              5100
0148                 ;;
0149                 ;; LOAD DATA FROM TAPE
0150                 ;;
0151  0107    3E01   LTAPE: MVI  A,01H
```



```
0152  0109    D3FA          OUT  0FAH      ; DMA INHIBIT
0153  010B    0E00          MVI  C,0       ; C=CHECKSUM REGISTER
0154  010D    CD            CALL CKSMI     ; DATA READ & CHECKSUM
              4901
0155  0110    67            MOV  H,A       ; H=START ADDRESS [HI]
0156  0111    CD            CALL CKSMI     ; DATA READ & CHECKSUM
              4901
0157  0114    6F            MOV  L,A       ; L=START ADDRESS [LO]
0158  0115    CD            CALL CKSMI     ; DATA READ & CHECKSUM
              4901
0159  0118    57            MOV  D,A       ; D=END ADDRESS [HI]
0160  0119    CD            CALL CKSMI     ; DATA READ & CHECKSUM
              4901
0161  011C    5F            MOV  E,A       ; E=END ADDRESS [LO]
0162  011D    22            SHLD ADRES     ; START ADDRESS STORE TO ADDRESS REG
              EE83
0163  0120    EB            XCHG
0164  0121    22            SHLD DATA      ; END ADDRESS STORE TO DATA REG
              EC83
0165  0124    EB            XCHG
0166  0125    2B            DCX  H
0167  0126    23     TAPE2: INX  H
0168  0127    CD            CALL CKSMI     ; DATA READ & CHECKSUM
              4901
0169  012A    77            MOV  M,A       ; DATA STORE TO MEMORY
0170  012B    7D            MOV  A,L
0171  012C    BB            CMP  E         ; START[LO],END[LO] COMPARE
0172  012D    C2            JNZ  TAPE2
              2601
0173  0130    7C            MOV  A,H
0174  0131    BA            CMP  D         ; START[HI],END[HI] COMPARE
0175  0132    C2            JNZ  TAPE2
              2601
0176  0135    CD            CALL CKSMI     ; DATA READ & CHECKSUM
              4901
0177  0138    C2            JNZ  ERROR     ; IF ZERO FLAG=ZERO--->CHECKSUM ERROR
              CB02
0178  013B    CD            CALL RGDSP
              A101
0179  013E    C3            JMP  START     ; END LOAD DATA
              5100
0180  0141    F5     CKSMO: PUSH PSW       ; PSW SAVE
0181  0142    81            ADD  C         ; CHECKSUM
0182  0143    4F            MOV  C,A
0183  0144    F1            POP  PSW       ; PSW RESTORE
0184  0145    CD            CALL SRIOT     ; DATA OUT
              7C02
0185  0148    C9            RET
0186  0149    CD     CKSMI: CALL SRIIN     ; DATA READ
              A002
0187  014C    47            MOV  B,A       ; ACC SAVE
```

```
0188 014D   81            ADD  C          ; CHECKSUM
0189 014E   4F            MOV  C,A
0190 014F   78            MOV  A,B        ; ACC RESTORE
0191 0150   C9            RET
0192                ;;
0193                ;; BREAK ENTRY
0194                ;;     BREAK & ONE STEP OPERATION
0195                ;;
0196 0151   E3      BRENT: XTHL           ; HL<-->PC(SAVED)
0197 0152   22            SHLD PSAVE      ; PC(LO) $83E0,PC(HI) $83E1 SAVED
            E083
0198 0155   F5            PUSH PSW        ; PSW SAVE
0199 0156   21            LXI  H,4H
            0400
0200 0159   39            DAD  SP         ; HL<--SP
0201 015A   F1            POP  PSW        ; PSW RECOVER
0202 015B   22            SHLD SSAVE      ; SP(LO) $83E2,SP(HI) $83E3 SAVED
            E283
0203 015E   E1            POP  H          ; HL RECOVER
0204 015F   31            LXI  SP,DATA
            EC83
0205 0162   F5            PUSH PSW        ; A $83EB,F $83EA SAVED
0206 0163   C5            PUSH B          ; B $83E9,C $83E8 SAVED
0207 0164   D5            PUSH D          ; D $83E7,E $83E6 SAVED
0208 0165   E5            PUSH H          ; H $83E5,L $83E4 SAVED
0209 0166   31            LXI  SP,MONSP   ; SP INITIALIZE
            0183
0210 0169   3A            LDA  BRKCT
            F283
0211 016C   A7            ANA  A          ; BREAK COUNTER=0 ?
0212 016D   CA            JZ   BSTOP      ; IF ZERO ONE STEP
            8B01                               LE83F0  HE83F1
0213 0170   2A            LHLD BRKAD
            F083
0214 0173   EB            XCHG            ; DE = BREAK POINTER
0215 0174   2A            LHLD PSAVE      ; HL = (PC)
            E083
0216 0177   7D            MOV  A,L
0217 0178   BB            CMP  E          ; PC[LO]-BP[LO]
0218 0179   C2            JNZ  NOBRK      ; IF NON ZERO NOT BREAK
            8501
0219 017C   7C            MOV  A,H
0220 017D   BA            CMP  D          ; PC[HI]-BP[HI]
0221 017E   C2            JNZ  NOBRK      ; IF NON ZERO NOT BREAK
            8501
0222 0181   21            LXI  H,BRKCT
            F283
0223 0184   35            DCR  M          ; BREAK COUNTER DECREMENT
0224 0185   CD      NOBRK: CALL ADDSP     ; (PSW) & (PC) DISPLAY
            9101
0225 0188   C3            JMP  RESRG      ; REGISTER RESTOR & RETURN
```



```
            F901
0226 018B   CD      BSTOP: CALL ADDSP     ; (PSW) & (PC) DISPLAY
            9101
0227 018E   C3            JMP  START
            5100
0228 0191   2A      ADDSP: LHLD FSAVE     ; HL=(PSW)
            EA83
0229 0194   22            SHLD DATA       ; (PSW)-->DATA REG
            EC83
0230 0197   2A            LHLD PSAVE      ; HL=(PC)
            E083
0231 019A   22            SHLD ADRES      ; (PC)-->ADDRESS REG
            EE83
0232 019D   CD            CALL RGDSP      ; ADDRESS & DATA REG DISPLAY
            A101
0233 01A0   C9            RET
0234                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0235                ;;            SUBROUTINE                    ;;
0236                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
0237                ;;
0238                ;;  ADDRESS,DATA REG DISPLAY
0239                ;;
0240 01A1   21      RGDSP: LXI  H,ADRES+1 ; HL=ADDRESS BUFFER ADDRESS
            EF83
0241 01A4   11            LXI  D,DISP     ; DE=DISPLAY DATA BUFFER ADDRESS
            F483
0242 01A7   0604          MVI  B,4        ; COUNTER SET
0243 01A9   7E            MOV  A,M
0244 01AA   12            STAX D          ; DISPLAY DATA STORE
0245 01AB   2B            DCX  H
0246 01AC   13            INX  D
0247 01AD   05            DCR  B
0248 01AE   C2            JNZ  $-5
            A901
0249 01B1   CD            CALL SEGCG      ; SEGMENT CONVERT
            C001
0250 01B4   C9            RET
0251                ;;
0252                ;;  DATA REG SHIFT (4 BITS)
0253                ;;
0254 01B5   2A      SHIFT: LHLD DATA
            EC83
0255 01B8   29            DAD  H
0256 01B9   29            DAD  H
0257 01BA   29            DAD  H
0258 01BB   29            DAD  H
0259 01BC   22            SHLD DATA
            EC83
0260 01BF   C9            RET
0261                ;;
0262                ;;  SEGMENT CONVERT SUB
```

```
0263                        ;;
0264  01C0   21             SEGCG: LXI  H,DISP     ; HL=DISPLAY DATA ADDRESS
             F483
0265  01C3   11                    LXI  D,DIG      ; DE=SEGMENT BUFFER ADDRESS
             F883
0266  01C6   01                    LXI  B,SEGD     ; BC=SEGMENT DATA ADDRESS
             E901
0267  01C9   7E                    MOV  A,M
0268  01CA   23                    INX  H
0269  01CB   E5                    PUSH H
0270  01CC   F5                    PUSH PSW
0271  01CD   E6F0                  ANI  OFOH       ; MASK 'FO'
0272  01CF   0F                    RRC
0273  01D0   0F                    RRC
0274  01D1   0F                    RRC
0275  01D2   0F                    RRC             ; SHIFT RIGHT 4 BITS
0276  01D3   2600                  MVI  H,0
0277  01D5   6F                    MOV  L,A
0278  01D6   09                    DAD  B
0279  01D7   7E                    MOV  A,M        ; A=SEGMENT DATA
0280  01D8   12                    STAX D          ; STORE SEGMENT DATA
0281  01D9   13                    INX  D
0282  01DA   F1                    POP  PSW
0283  01DB   E60F                  ANI  OFH        ; MASK 'OF'
0284  01DD   2600                  MVI  H,0
0285  01DF   6F                    MOV  L,A
0286  01E0   09                    DAD  B
0287  01E1   7E                    MOV  A,M        ; A=SEGMENT DATA
0288  01E2   12                    STAX D          ; STORE SEGMENT DATA
0289  01E3   E1                    POP  H
0290  01E4   1C                    INR  E
0291  01E5   C2                    JNZ  SEGCG+9
             C901
0292  01E8   C9                    RET
0293                        ;;
0294                        ;;  SEGMENT DATA  ;;
0295                        ;;
0296  01E9   5C06           SEGD:  DB   5CH,06H,5BH,4FH,66H,6DH
             5B4F
             666D
0297  01EF   7D27                  DB   7DH,27H,7FH,6FH,77H,7CH,39H
             7F6F
             777C
0298  01F6   39                    DB   5EH,79H,71H
             5E
             7971
0299                        ;;
0300                        ;; REGISTER RESTORE
0301                        ;;
0302  01F9   2A             RESRG: LHLD SSAVE
             E283
```



```
0303  01FC   F9                    SPHL            ; SP RESTORE
0304  01FD   2A                    LHLD PSAVE
             E083
0305  0200   E5                    PUSH H          ; PC STORED IN USER STACK
0306  0201   2A                    LHLD LSAVE
             E483
0307  0204   E5                    PUSH H          ; HL STORED BELOW USER STACK
0308  0205   2A                    LHLD FSAVE
             EA83
0309  0208   E5                    PUSH H          ; PSW STORED BELOW USER STACK
0310  0209   2A                    LHLD CSAVE
             E883
0311  020C   4D                    MOV  C,L
0312  020D   44                    MOV  B,H        ; BC RESTORED
0313  020E   2A                    LHLD ESAVE
             E683
0314  0211   EB                    XCHG            ; DE RESTORED
0315  0212   F1                    POP  PSW        ; PSW RESTORED
0316  0213   E1                    POP  H          ; HL RESTORED
0317  0214   FB                    EI              ; INTERRUPT ENABLE
0318  0215   C9                    RET             ; PC RESTORED & GO TO USER CONTROL
                                                     ROUTINE
0319                        ;;
0320                        ;; KEY INPUT
0321                        ;;
0322                        ;;   ACC=INPUT DATA
0323                        ;;
0324  0216   CD             KEYIN: CALL INPUT      ; KEY INPUT
             2302
0325                        ; KEY IN---> ACC=DATA  & KFLAG SET
0326                        ; NON IN---> ACC=FF    & KFLAG RESET
0327  0219   47                    MOV  B,A        ; INPUT DATA SAVE
0328  021A   3A                    LDA  KFLAG      ; ACC=KEY FLAG
             F383
0329  021D   A7                    ANA  A
0330  021E   CA                    JZ   KEYIN      ; IF ZERO JMP KEYIN
             1602
0331  0221   78                    MOV  A,B        ; INPUT DATA RESTORE
0332  0222   C9                    RET
0333                        ;;
0334                        ;; KEY INPUT SUB
0335                        ;;
0336  0223   CD             INPUT: CALL KEY        ; KEY SCAN
             4702
0337  0226   3C                    INR  A
0338  0227   CA                    JZ   NOKEY      ; JMP NON INPUT
             4202
0339  022A   CD                    CALL D2         ; WAIT CHATTERING TIME
             EA02
0340  022D   CD                    CALL KEY        ; KEY SCAN
             4702
0341  0230   47                    MOV  B,A        ; INPUT DATA SAVE
```

```
0342  0231  3C              INR   A
0343  0232  CA              JZ    NOKEY      ; JMP NON INPUT
            4202
0344  0235  3A              LDA   KFLAG      ; A=KEY INPUT FLAG
            F383
0345  0238  A7              ANA   A
0346  0239  C2              JNZ   $-15       ; JMP IF KEY FLAG IS SET
            2A02
0347  023C  3D              DCR   A          ; A=FFH
0348  023D  32              STA   KFLAG      ; SET KEY INPUT FLAG
            F383
0349  0240  78              MOV   A,B        ; INPUT DATA RESTORE
0350  0241  C9              RET
0351  0242  06FF    NOKEY:  MVI   B,0FFH
0352  0244  C3              JMP   $-7
            3D02
0353                ;;
0354                ;;  KEY SCAN & CONVERT HEXA DATA SUB
0355                ;;
0356  0247  1600    KEY:    MVI   D,0
0357  0249  42              MOV   B,D
0358  024A  3EEF            MVI   A,0EFH
0359  024C  D3FA            OUT   0FAH       ; PORT C SCAN LIN DATA SET
0360  024E  DBF8            IN    0F8H       ; KEY SCAN 0--->7
0361  0250  EEFF            XRI   0FFH       ; COMPLEMENT
0362  0252  C2              JNZ   KEYI
            7102
0363  0255  0608            MVI   B,8
0364  0257  3EDF            MVI   A,0DFH
0365  0259  D3FA            OUT   0FAH       ; PORT C SCAN LIN DATA SET
0366  025B  DBF8            IN    0F8H       ; KEY SCAN 8--->F
0367  025D  EEFF            XRI   0FFH       ; COMPLEMENT
0368  025F  C2              JNZ   KEYI
            7102
0369  0262  0610            MVI   B,10H
0370  0264  3EBF            MVI   A,0BFH
0371  0266  D3FA            OUT   0FAH       ; PORT C SCAN LINE DATA SET
0372  0268  DBF8            IN    0F8H       ; KEY SCAN FUNCTION
0373  026A  EEFF            XRI   0FFH       ; COMPLEMENT
0374  026C  C2              JNZ   KEYI
            7102
0375  026F  3D              DCR   A          ; A=FFH
0376  0270  C9              RET
0377  0271  0F      KEYI:   RRC
0378  0272  DA              JC    $+7        ; DATA=X--BIT?
            7902
0379  0275  14              INR   D
0380  0276  C3              JMP   KEYI
            7102
0381  0279  7A              MOV   A,D
0382  027A  B0              ORA   B          ; SCAN LINE--->DATA MODIFY
```



```
0383  027B  C9              RET
0384                ;;
0385                ;;  SRIAL OUT PUT ROUTINE
0386                ;;
0387  027C  D5      SRIOT:  PUSH  D          ; DE REGISTER SAVE
0388  027D  C5              PUSH  B          ; BC REGISTER SAVE
0389  027E  0608            MVI   B,8        ; BIT COUNTER SET
0390  0280  4F              MOV   C,A        ; C<--ACC
0391  0281  AF              XRA   A          ; SET START BIT
0392  0282  D3FA            OUT   0FAH       ; OUT PUT START BIT
0393  0284  CD              CALL  D2         ; WAIT 1 BIT TIME
            EA02
0394  0287  79      SRI01:  MOV   A,C        ; ACC<--DATA
0395  0288  E67F            ANI   7FH        ; MASK M.S.B.
0396  028A  D3FA            OUT   0FAH       ; OUTPUT L.S.B.
0397  028C  79              MOV   A,C
0398  028D  1F              RAR              ; DATA SHIFT
0399  028E  4F              MOV   C,A
0400  028F  CD              CALL  D2         ; WAIT 1 BIT TIME
            EA02
0401  0292  05              DCR   B          ; BIT COUNTER DECREMENT
0402  0293  C2              JNZ   SRI01      ; GO DO IT AGAIN
            8702
0403  0296  3E01            MVI   A,1        ; SET STOP BIT
0404  0298  D3FA            OUT   0FAH       ; OUTPUT STOP BIT
0405  029A  CD              CALL  D3         ; WAIT WORD INTERVAL
            EF02
0406  029D  C1              POP   B          ; BC REGISTER RESTORE
0407  029E  D1              POP   D          ; DE REGISTER RESTORE
0408  029F  C9              RET
0409                ;;
0410                ;;  SRIAL INPUT ROUTINE
0411                ;;
0412  02A0  D5      SRIIN:  PUSH  D          ; DE REGISTER SAVE
0413  02A1  C5              PUSH  B          ; BC REGISTER SAVE
0414  02A2  01              LXI   B,800H     ; REGISTER INITIALIZE
            0008
0415  02A5  DBF9    SRII1:  IN    0F9H       ; GET INPUT DATA
0416  02A7  1F              RAR              ; CHECK L.S.B.
0417  02A8  DA              JC    SRII1      ; JUMP BACK IF ZERO
            A502
0418  02AB  CD              CALL  D1         ; WAIT 1/2 BIT TIME
            DD02
0419  02AE  DBF9            IN    0F9H       ; GET INPUT DATA
0420  02B0  1F              RAR              ; CHECK L.S.B.
0421  02B1  DA              JC    SRII1      ; IF ONE START OVER
            A502
0422  02B4  CD              CALL  D2         ; WAIT 1BIT TIME
            EA02
0423  02B7  DBF9            IN    0F9H       ; GET INPUT DATA
0424  02B9  E601            ANI   1          ; MASK OUT L.S.B.
```

```
0425  02BB  81           ADD  C         ; ADD C TO ACC
0426  02BC  0F           RRC            ; DATA SHIFT
0427  02BD  4F           MOV  C,A       ; ACC SAVE TO C REG
0428  02BE  05           DCR  B         ; BIT COUNTER DECREMENT
0429  02BF  C2           JNZ  $-11      ; GO BACK IF NOT END
            B402
0430  02C2  CD           CALL D2        ; WAIT 1 BIT TIME
            EA02
0431  02C5  CD           CALL D2        ; WAIT 1 BIT TIME
            EA02
0432  02C8  C1           POP  B         ; BC REGISTER RESTORE
0433  02C9  D1           POP  D         ; DE REGISTER RESTORE
0434  02CA  C9           RET
0435  02CB  21    ERROR: LXI  H,DIG     ; HL=SEGMENT DATA BUFFER ADDRES
            F883
0436  02CE  3679         MVI  M,79H
0437  02D0  23           INX  H
0438  02D1  3680         MVI  M,80H
0439  02D3  2C           INR  L
0440  02D4  C2           JNZ  $-3       ; WRITE ERROR MESSAGE
            D102
0441  02D7  31           LXI  SP,MONSP  ; SP INITIALIZE
            D183
0442  02DA  C3           JMP  START+10
            5B00
0443              ;;
0444              ;;  BIT TIMER & CHATTERING TIMER
0445              ;;
0446  02DD  1624  D1:    MVI  D,24H     ; WAIT 1/2 BIT TIME  4.5112 MSEC
0447  02DF  1E0C         MVI  E,0CH
0448  02E1  1D           DCR  E
0449  02E2  C2           JNZ  $-1
            E102
0450  02E5  15           DCR  D
0451  02E6  C2           JNZ  $-7
            DF02
0452  02E9  C9           RET
0453  02EA  1648  D2:    MVI  D,48H     ; WAIT 1 BIT TIME  9.0176 MSEC
0454  02EC  C3           JMP  D1+2
            DF02
0455  02EF  16D8  D3:    MVI  D,0D8H    ; WAIT 3 BIT TIME  27.0176 MSEC
0456  02F1  C3           JMP  D1+2
            DF02
0457              ;;
0458              ;; ADDRESS TABEL
0459              ;;
0460              LSAVE: EQU  83E4H
0461              HSAVE: EQU  83E5H
0462              ESAVE: EQU  83E6H
0463              DSAVE: EQU  83E7H
0464              CSAVE: EQU  83E8H
```



```
0465              BSAVE: EQU  83E9H
0466              FSAVE: EQU  83EAH
0467              ASAVE: EQU  83EBH
0468              PSAVE: EQU  83E0H
0469              SSAVE: EQU  83E2H
0470              ADRES: EQU  83EEH
0471              DATA:  EQU  83ECH
0472              BRKAD: EQU  83F0H
0473              BRKCT: EQU  83F2H
0474              KFLAG: EQU  83F3H
0475              DISP:  EQU  83F4H
0476              DIG:   EQU  83F8H
0477              MONSP: EQU  83D1H
0478              USESP: EQU  83C7H
0479              RST2:  EQU  83D1H
0480              RST3:  EQU  83D4H
0481              RST4:  EQU  83D7H
0482              RST5:  EQU  83DAH
0483              RST6:  EQU  83DDH
0484                     END
```

```
*****   UCOM - 8    ASSEMBL    LIST   *****                    P 0001

  MONST   003B     START   0051     RST2    83D1     RST3    83D4
  RST4    83D7     RST5    83DA     RST6    83DD     BRENT   0151
  DATA    83EC     USESP   83C7     SSAVE   83E2     MONSP   83D1
  SEGCG   01C0     KEYIN   0216     DIGIT   0084     TABL    0074
  GOTO    00CC     RESRG   01F9     ADSET   0094     ADDCX   00B8
  ADINX   009D     MEMW    00C2     STAPE   00D5     LTAPE   0107
  SHIFT   01B5     RGDSP   01A1     ADRES   83EE     MEMR    00AD
  ADSTR   00A4     PSAVE   83E0     CKSM0   0141     TAPE1   00EF
  CKSMI   0149     TAPE2   0126     ERROR   02CB     SRI0T   027C
  SRIIN   02A0     BRKCT   83F2     BSTOP   018B     BRKAD   83F0
  NOBRK   0185     ADDSP   0191     FSAVE   83EA     DISP    83F4
  DIG     83F8     SEGD    01E9     LSAVE   83E4     CSAVE   83E8
  ESAVE   83E6     INPUT   0223     KFLAG   83F3     KEY     0247
  NOKEY   0242     D2      02EA     KEYI    0271     SRI01   0287
  D3      02EF     SRII1   02A5     D1      02DD     HSAVE   83E5
  DSAVE   83E7     BSAVE   83E9     ASAVE   83EF
```

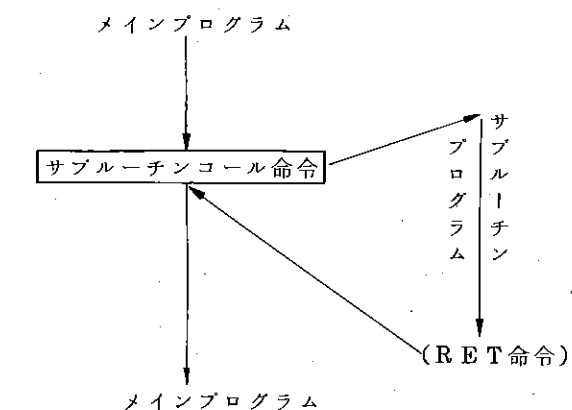

第4章　モニタ・サブルーチン

4.1　概　　要

モニタプログラムは，メインプログラムの他かにいくつかのサブルーチンプログラムで構成されています．

サブルーチンプログラムとは，繰り返し行われる処理や，だれもが行うような処理等を１つの一連のプログラムとしてまとめたものです．

プログラムを組む時に，次に述べるような処理が必要なとき，プログラムにサブルーチンコール命令を書くだけで，簡単サブルーチンプログラムを呼び出して使用することができます．

図４－１　サブルーチンのコール

この説明書では，ＴＫ－８０モニタプログラムを構成しているサブルーチンのうち，一般性のあるもの，特にアプリケーションプログラムを書く上で有効であると思われるものについて，その機能および入出力条件について説明します．

4.2　サブルーチンの考え方

メインプログラムからサブルーチンをコールした場合，サブルーチンでの処理が終了した場合のもどり番地（リターンアドレス）を記憶しておくことが必要です．

このもどり番地は，サブルーチンがコールされたときに，スタックポインタが指しているプッシュ・ダウン・スタックに自動的に書き込まれて，サブルーチンの終わりでＲＥＴ（リターン）命令を実行すると，引用されてもとのプログラムに戻ることができます．なおユーザプログラムの中でスタックポインタを操作する命令を使った場合には別の注意が必要です．

プログラムを RUN キーで走らせた直後のスタックポインタは83C7にセットされています．

またサブルーチンをコールする場合，その入出力条件というものを常に考えておかなければなりません．

つまりサブルーチンを実行する際，サブルーチン内で使うデータのセット方法やサブルーチン内で処理された結果得られたデータのセット状態，またサブルーチン内で値が破壊されてしまうレジスタは何か，というようなことを調べた上で必要なデータを得るとともに，メインルーチンで必要なデータがサブルーチン内で破壊されないような対策をこうずる必要があります．

図4－2　サブルーチン・コールの手続き例

```
LXI   SP,83C7H     スタックエリア確保
PUSH  PSW          レジスタセーブ
PUSH  B
MVI   A,0          パラメータのセット
MVI   B,8
CALL  SUBROUTINE   サブルーチン・コール
POP   B            レジスタ復帰
POP   PSW
```

サブルーチン・プログラム

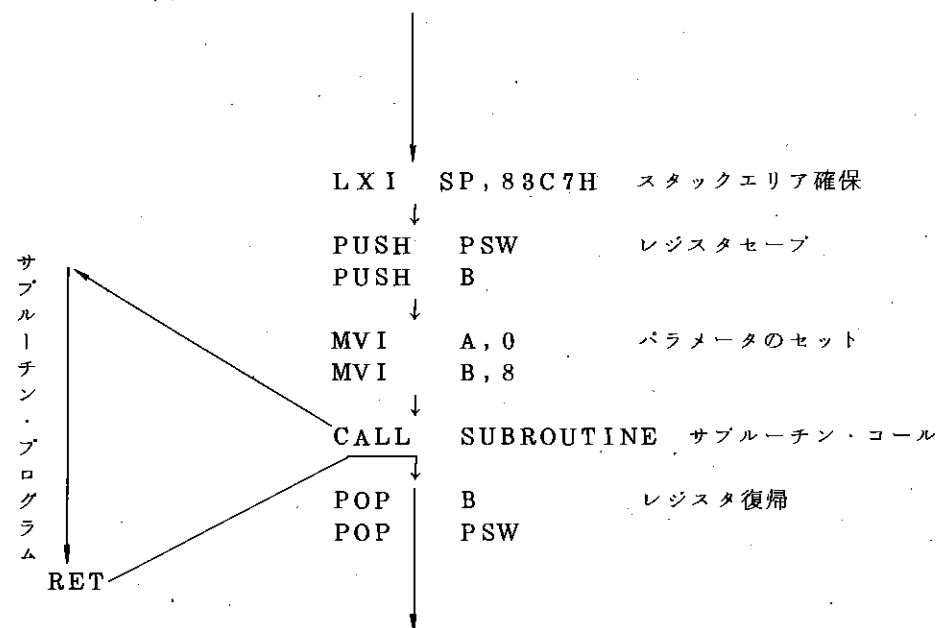
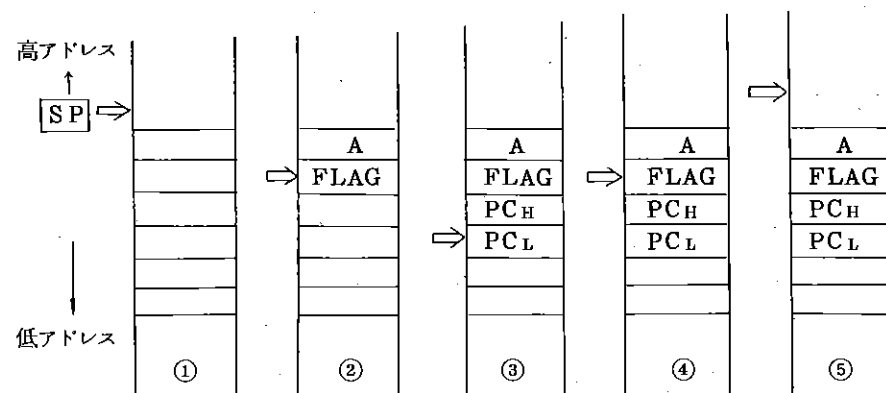
RET

(1) 基本的なスタック操作の例

高アドレス

SP

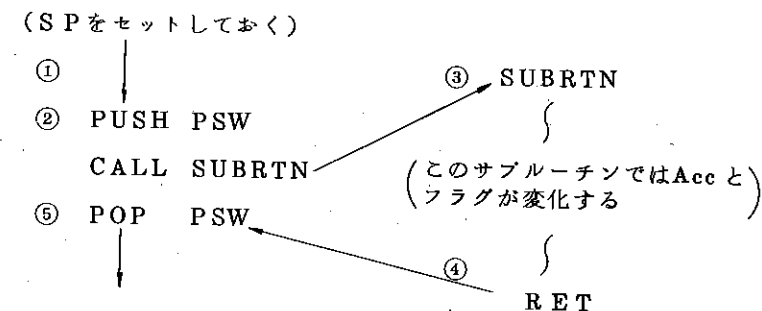
低アドレス

①	②	③	④	⑤
	A	A	A	A
	FLAG	FLAG	FLAG	FLAG
		PCH	PCH	PCH
		PCL	PCL	PCL

① SPはスタックの開始アドレスを示しています．

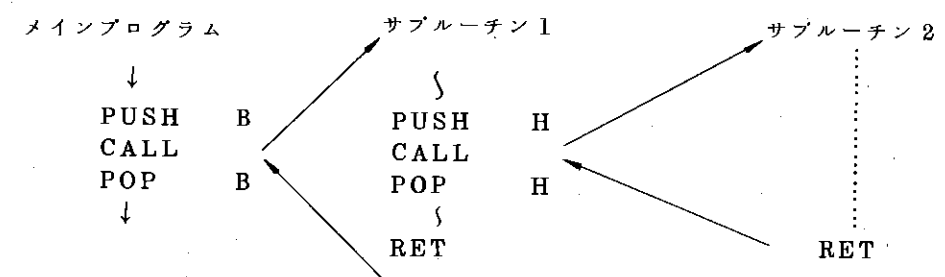
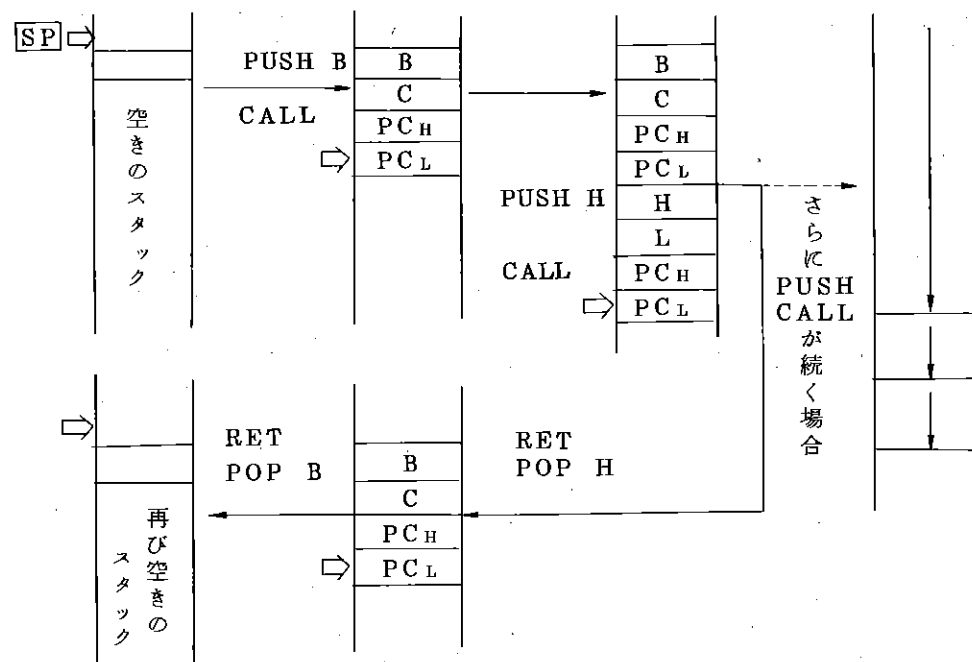
② Accとフラグがスタックに書かれ，SPは（－2）されます．

③ 戻り番地がスタックに書かれ，SPは（－2）されます．

④ 戻り番地が引用され，SPは（＋2）されます．

⑤ Accとフラグが復帰され，SPは（＋2）されます．

このプログラムでは，サブルーチン処理した後もAccとフラグの内容を破壊させないために，サブルーチンに入る直前にスタックに退避させ，戻ってからすぐ復帰させています．ここでスタックポインタは，図の①～⑤のように動きます．このような動きをするスタックをプッシュ・ダウン・スタックと呼びます．

μPD8080Aでは，スタックに書き込む（PUSH命令を実行する）とスタックポインタ(SP)の値は自動的に2番地デクリメントされ，スタックから読み出す（POP命令を実行する）と，逆に2番地インクリメントされます．従ってPUSH命令やCALL命令を続けて実行させると，スタックの先頭番地はどんどん下がってきます（スタックが深くなるといいます）が，POP，RET命令を同じ回数だけ実行するとスタックの先頭アドレスは元に戻り，スタックの内容は空になります．この様子を図示すると図4－3のようになります．

図4－3

4.3 サブルーチンの機能説明

次に示す７つのサブルーチンについてこれから説明します．　　スタート番地

- セグメントデータ変換サブルーチン　　01 C0
- アドレスレジスタ，データレジスタ表示サブルーチン　　01 A1
- キー入力サブルーチン(1)　　02 23
- キー入力サブルーチン(2)　　02 16
- シリアル出力サブルーチン　　02 7C
- シリアル入力サブルーチン　　02 A0
- タイマ・サブルーチン　　D1 4.5112mS 02 DD / D2 9.0176mS 02 EA / D3 27.0176mS 02 EF

4.3.1 セグメントデータ変換サブルーチン

(1) サブルーチン名

SEGCG

(2) スタート番地

０１Ｃ０番地

(3) 入出力条件

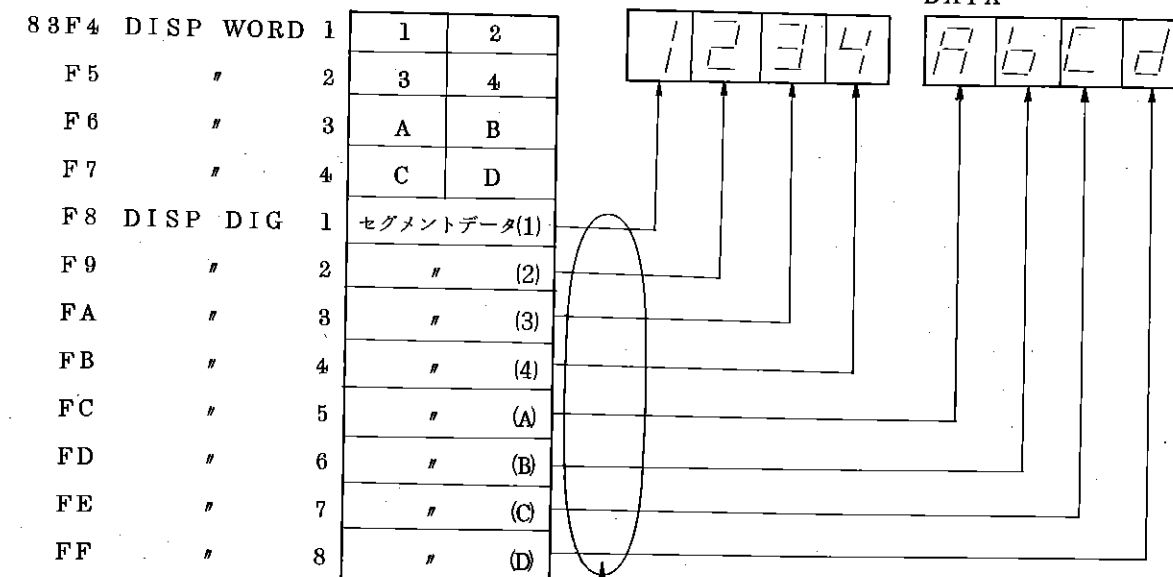
入力パラメータ	なし
出力パラメータ	なし
使用レジスタ	A，F，B，C，D，E，H，L
使用スタック	2レベル

(4) 機　能

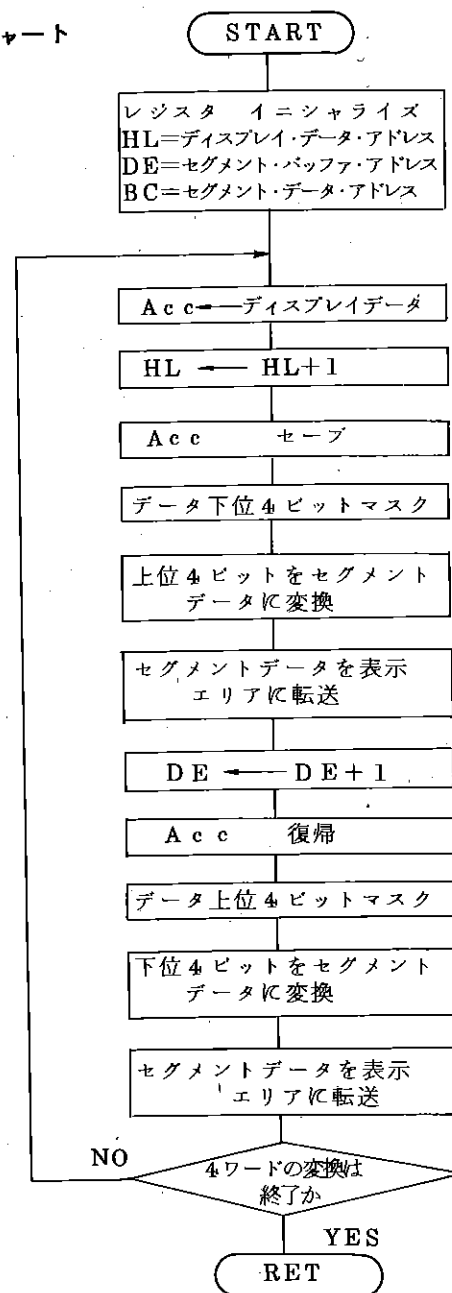
ディスプレイ・レジスタ・エリアの４ワードに格納されているデータの上位４ビット，下位４ビットを各々１６進とみなして，ＬＥＤの７セグメントデータに変換し，セグメント・データ・バッファに転送します．

従って１６進数をＬＥＤに表示させたい場合には，表示したい数値をディスプレイ・レジスタ・エリアにセットしてから，このサブルーチンをコールするだけですみます．

各レジスタおよびＬＥＤディスプレイにおけるデータ転送の関係は，次のようになっています．

(ディスプレイ・レジスタ)	データ変換	(セグメント・データ・バッファ)
DISP WORD1〔上位４ビット〕	………→	DISP DIG1
DISP WORD1〔下位４ビット〕	………→	DISP DIG2
DISP WORD2〔上位４ビット〕	………→	DISP DIG3
DISP WORD2〔下位４ビット〕	………→	DISP DIG4
DISP WORD3〔上位４ビット〕	………→	DISP DIG5
DISP WORD3〔下位４ビット〕	………→	DISP DIG6
DISP WORD4〔上位４ビット〕	………→	DISP DIG7
DISP WORD4〔下位４ビット〕	………→	DISP DIG8

セグメントの組み合わせに変換されたデータは，１桁ずつ周期的に表示回路に送られＬＥＤをダイナミック点灯しています．

ＴＫ－８０ではＤＭＡ（ダイレクト・メモリ・アクセス）転送という方式を用いていますので，この処理はプログラムを書かなくても自動的に行われます．

(5) フローチャート

(6) 使　用　例

```
8200  MVI   A,1      3E01     ┐
8202  STA   83F4H    32F483   │
8205  MVI   A,23H    3E23     │
8207  STA   83F5H    32F583   │ ①
820A  MVI   A,45H    3E45     │
820C  STA   83F6H    32F683   │
820F  MVI   A,67H    3E67     │
8211  STA   83F7H    32F783   ┘
8214  CALL  SEGCG    CDC001     ②
8217  HLT            76
```


① 表示したいデータをディスプレイ・レジスタにセットします。
② 変換ルーチンをコールします。

このプログラムでLED表示部には，01234567 が表示されます。

備考　データの種類が何であっても，必ず0～Fまでの16進数として表示されます。

4.3.2 アドレスレジスタ，データレジスタ表示サブルーチン

(1) サブルーチン名

RGDSP

(2) スタート番地

01A1番地

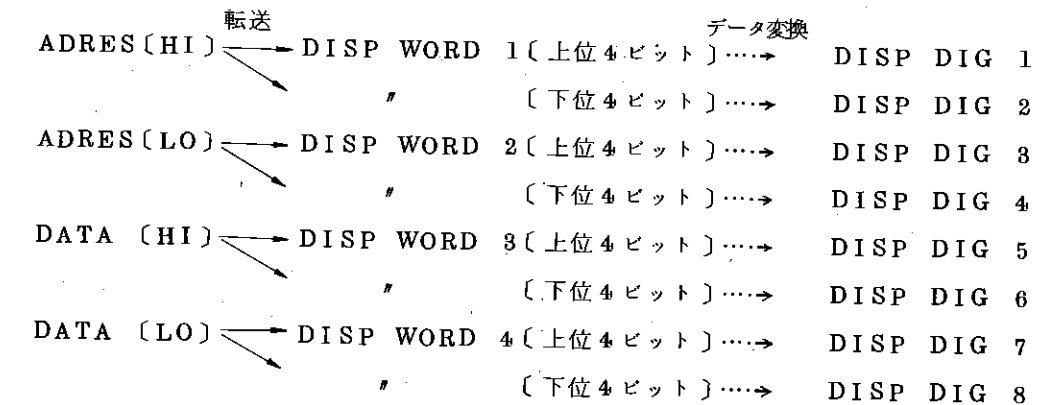
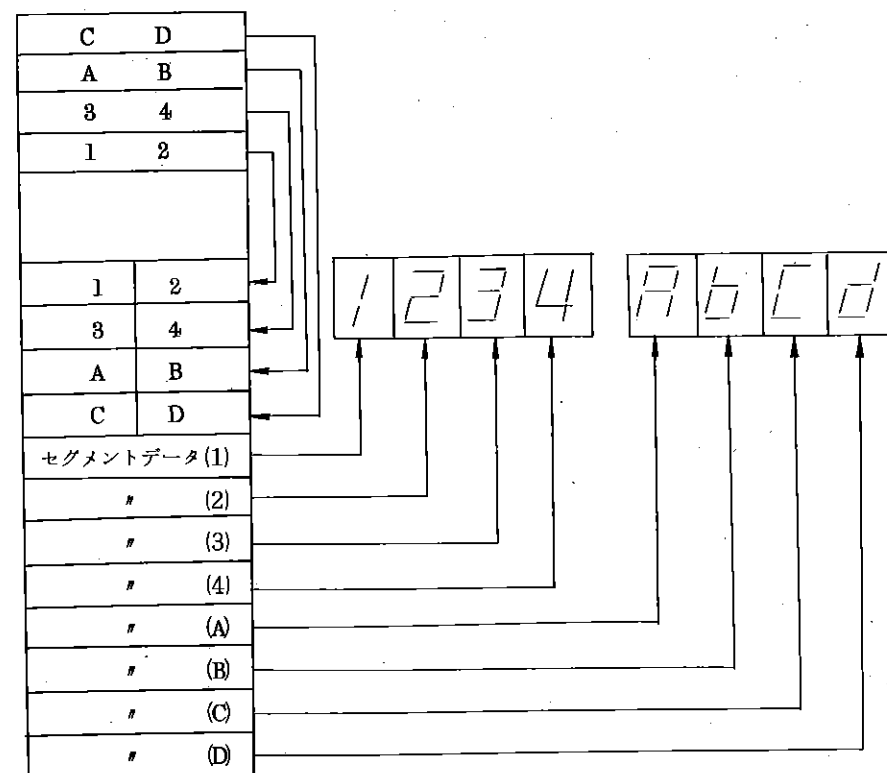
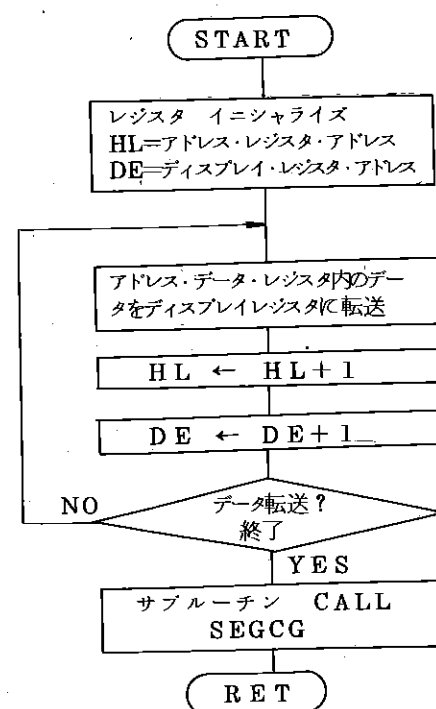
(3) 入出力条件

入力パラメータ	なし
出力パラメータ	なし
使用レジスタ	A，F，B，C，D，E，H，L
使用スタック	3レベル

(4) 機　　能

アドレスレジスタおよびデータレジスタにセットされているデータをディスプレイ・レジスタに転送し，さらに各データの上位4ビット，下位4ビットを各々16進数として各々セグメントデータに変換して，セグメント・データ・バッファに転送します。

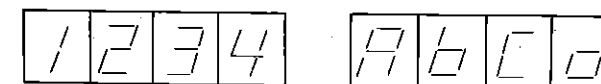
これにより，アドレスレジスタおよびデータレジスタにセットされているデータを，LEDディスプレイに表示させることができます。

各レジスタおよびLEDディスプレイにおけるデータ転送の関係は，次のようになっています。

```
            転送                          データ変換
ADRES〔HI〕──→ DISP WORD 1〔上位4ビット〕…→  DISP DIG 1
          ↘       〃      〔下位4ビット〕…→  DISP DIG 2
ADRES〔LO〕──→ DISP WORD 2〔上位4ビット〕…→  DISP DIG 3
          ↘       〃      〔下位4ビット〕…→  DISP DIG 4
DATA 〔HI〕──→ DISP WORD 3〔上位4ビット〕…→  DISP DIG 5
          ↘       〃      〔下位4ビット〕…→  DISP DIG 6
DATA 〔LO〕──→ DISP WORD 4〔上位4ビット〕…→  DISP DIG 7
          ↘       〃      〔下位4ビット〕…→  DISP DIG 8
```

(5) フローチャート

(6) 使用例

HLレジスタの内容をアドレス・ディスプレイに，DEレジスタの内容をデータ・ディスプレイに表示します．


```
8200  LXI   H,1234H   21 34 12
      LXI   D,0ABCDH  11 CD AB
      SHLD  ADRES     22 EE 83
      XCHG            EB
      SHLD  DATA      22 EC 83
      CALL  RGDSP     CD A1 01
      HLT             76
```

ADRES ： 83EE番地
DATA ： 83EC番地
RGDSP ： 01A1番地

LEDディスプレイの表示は次のようになります．

4.3.3 キー入力サルブーチン（1）

(1) サブルーチン名

INPUT

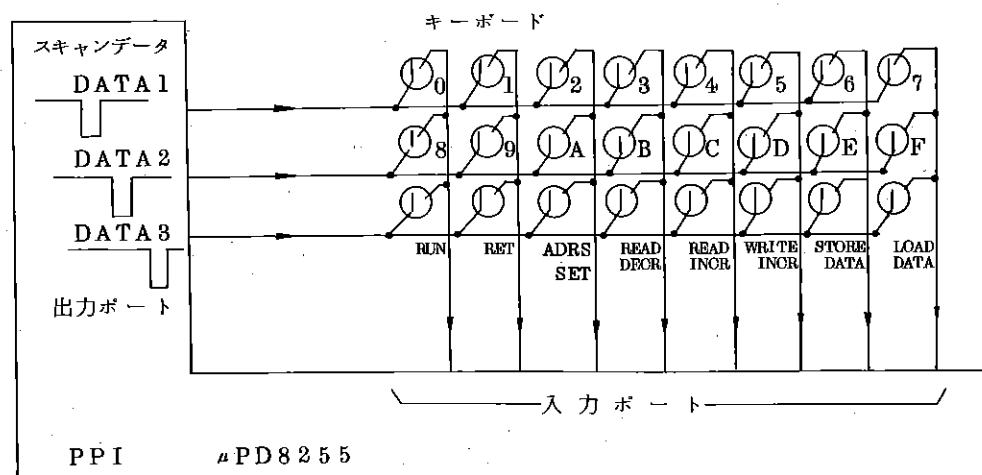
(2) スタート番地

0223番地

(3) 入出力条件

入力パラメータ	なし	
出力パラメータ	キー入力なし	A=FF
		キークラブ=00（83F3番地）
	キー入力あり	A=入力データ〔HEXA〕
		キークラブ=FF（83F3番地）
使用レジスタ	A，F，B，D，E	
使用スタック	1レベル	

(4) 機　能

キー入力があるかどうかを，すべてのキーボードスイッチについて1回ずつ調べます．(実際は次の図のように3つのブロックごとに8キー同時に調べます)．

1回のスキャンニングにおいて，どのキーボード・スイッチも押されていない場合は，キーフラグをリセットした後，アキュムレータにデータ"FF"をセットし，リターンします.

1回のスキャンニングにおいて，キーボード・スイッチが押されていることが検出された場合，キーフラグをセンスして，新しく押されたキーかどうかを調べます.

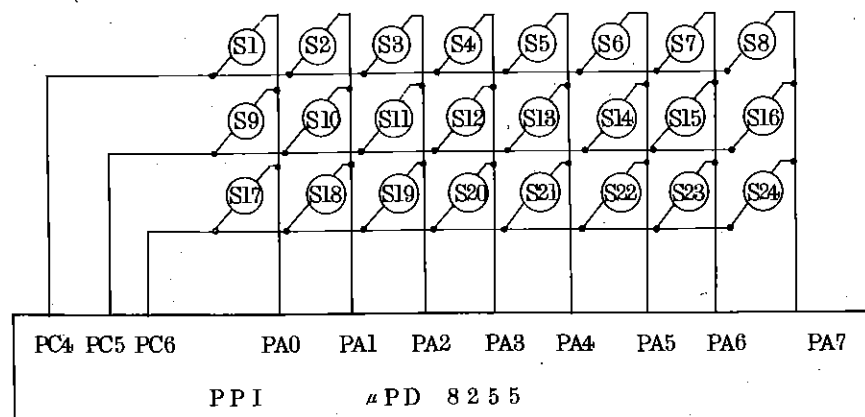
もしこれが前から押し続けられているものであった場合，離されるまで待ち，離されるとキーフラグをリセットした後アキュムレータにデータ"FF"をセットしてリターンします.

またこれが今回はじめて押されたものである場合は，チャタリング・タイマの設定時間だけ待ち合わせた後，アキュムレータにはキー入力の位置に対応した16進データをセットし，キーフラグをセットした後リターンします.

このサブルーチンを使用すれば，プログラムに何らかの処理を実行させながら，キーの状態をモニタすることができます.

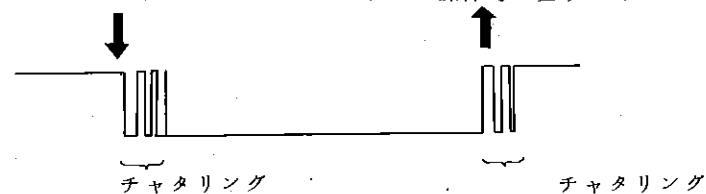
キーボードをセンスした場合，CPUにはキーが押されていない場合は，"FF"（すべてのビットに"1"が立っている），キーが押されている場合は，押されたキーに対応するビットだけが，"0"となっているデータが読み込まれます.

本サブルーチンでは，キーが押されていることが検出されると，そのキーに対応する16進データに変換します.

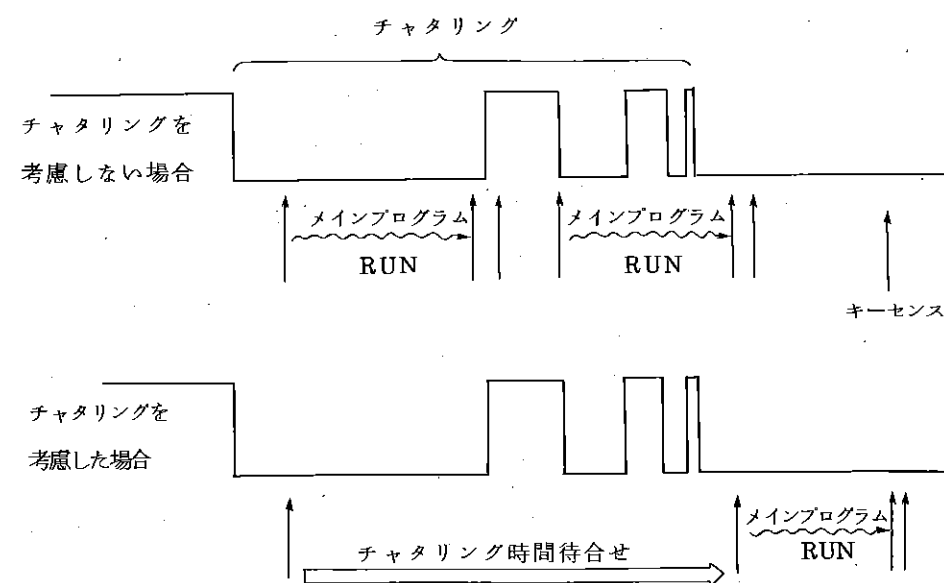
キーボードスイッチと変換されるデータとの関係は次のようになります.

	DIGIT & FUNCTION	HEXA DATA
S1	0	00
2	1	01
3	2	02
4	3	03
5	4	04
6	5	05
7	6	06
8	7	07
9	8	08
10	9	09
11	A	0A
12	B	0B
13	C	0C
14	D	0D
15	E	0E
16	F	0F
17	RUN	10
18	RET	11
19	ADRS SET	12
20	READ DECR	13
21	READ INCR	14
22	WRITE INCR	15
23	STORE DATA	16
24	LOAD DATA	17

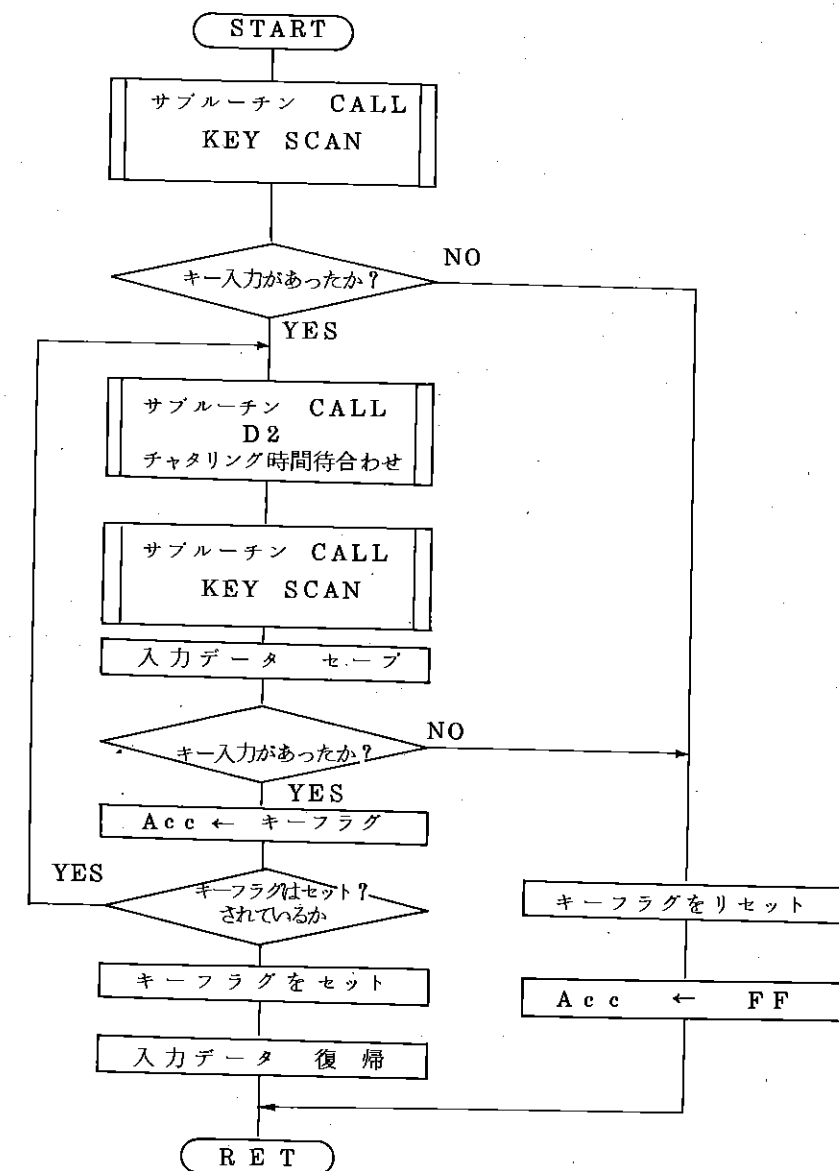
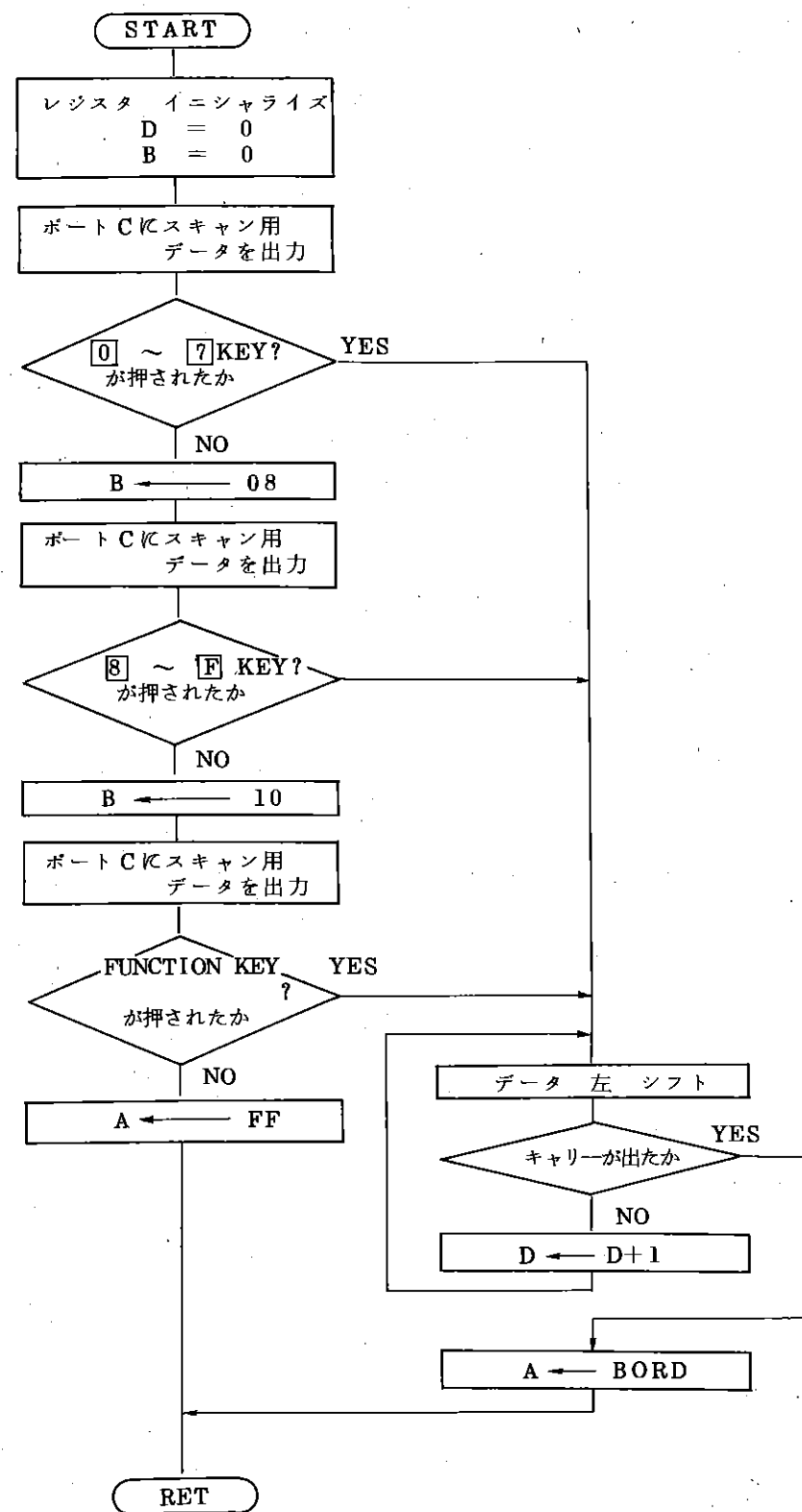
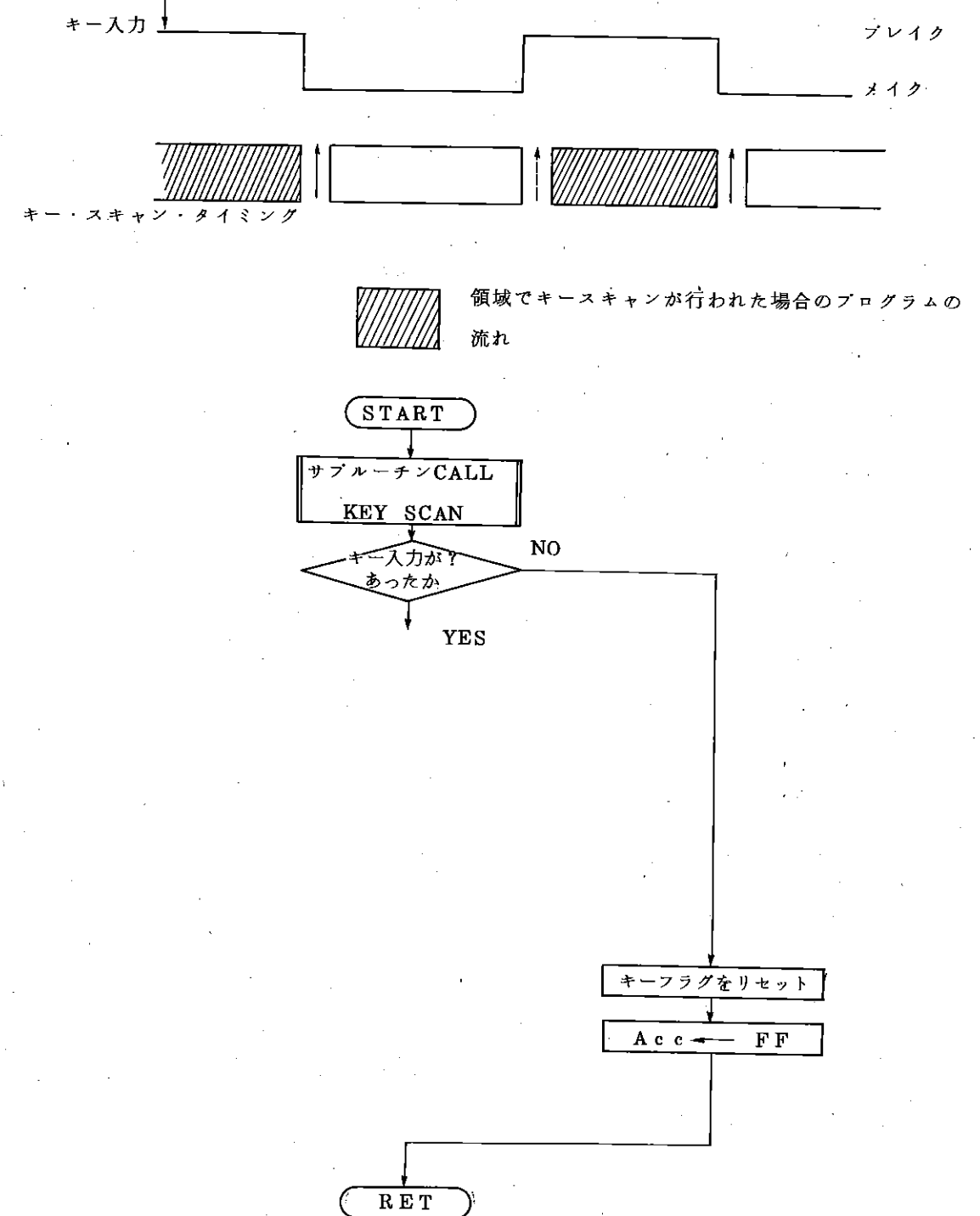
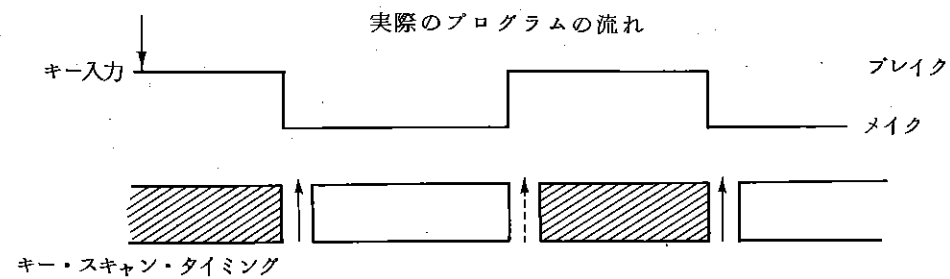
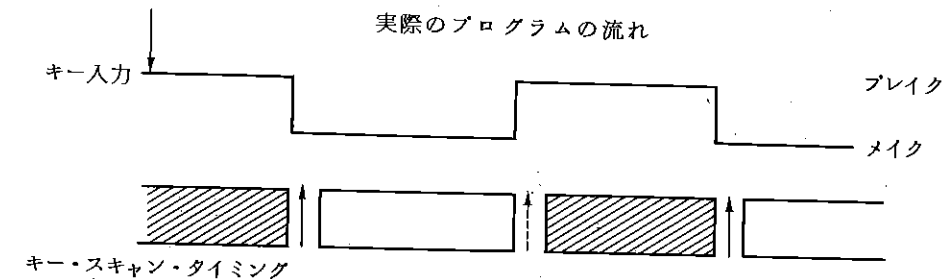
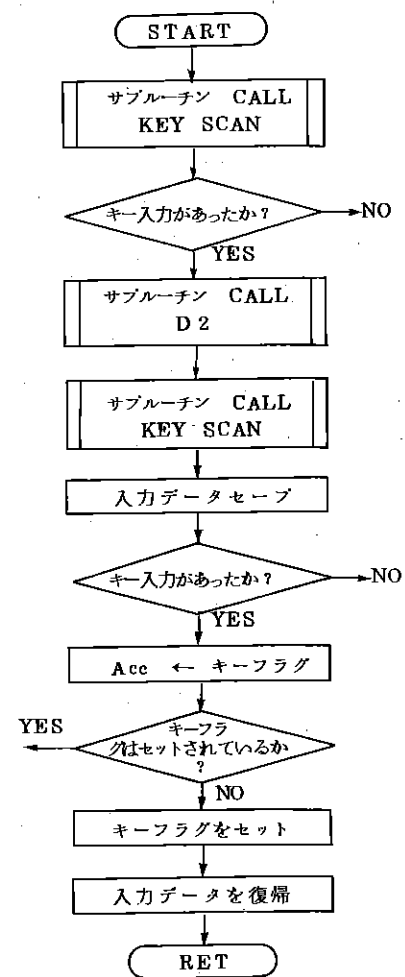
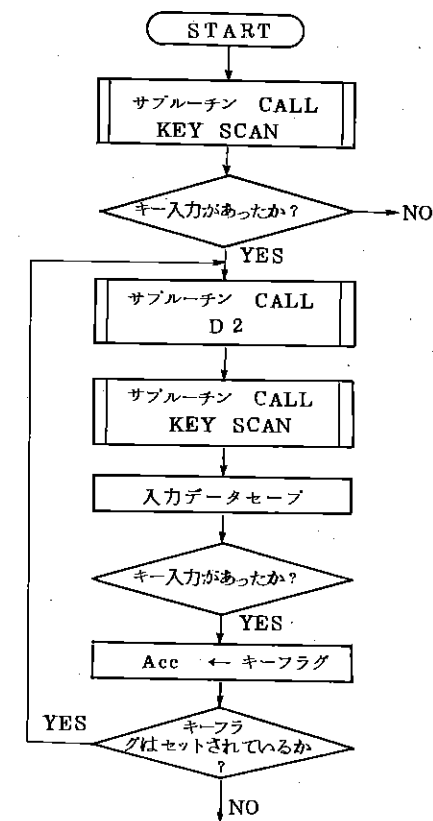
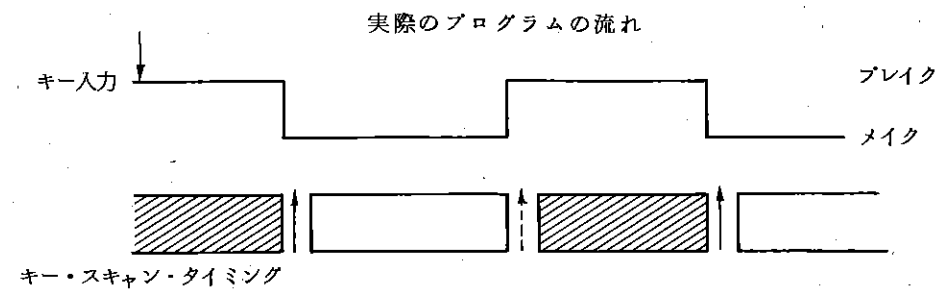
備考　キースイッチに付けられている数字とファンクション名は，モニタプログラムで解釈される意味に合わせてありますが，キーの位置とその意味付けは，プログラムの処理のみで決まるものですから，同じキーに違った意味をもたせて使用することもできます.

キー・チャタリングをどのように処理しているか

メカニカルタイプのスイッチでは，その操作時に必ずチャタリングが生じます．

本キットで使用しているキーボードスイッチのチャタリング時間は，最大５msec となっています．

このサブルーチンでは，このチャタリングを考慮してキー入力を検出すると，チャタリングの時間を待ち合わせ，もう一度キーセンスを行うことによって，チャタリング時間中にプログラムが走り出さないようになっています．

これを行わない場合，次の図のように一度しかキーを押さないのに，２度以上動作を行ってしまう可能性があります．

(5) フローチャート

キースキャンサブルーチン　KEY SCAN
START
レジスタ　イニシャライズ
D = 0
B = 0
ポートCにスキャン用データを出力
0 ～ 7 KEY? が押されたか
YES
NO
B ← 08
ポートCにスキャン用データを出力
8 ～ F KEY? が押されたか
NO
B ← 10
ポートCにスキャン用データを出力
FUNCTION KEY? が押されたか
YES
NO
A ← FF
データ　左　シフト
キャリーが出たか
YES
NO
D ← D+1
A ← BORD
RET
実際のプログラムの流れ
キー入力
ブレイク
メイク
キー・スキャン・タイミング
領域でキースキャンが行われた場合のプログラムの流れ
START
サブルーチンCALL
KEY SCAN
キー入力が？あったか
NO
YES
キーフラグをリセット
Acc ← FF
RET

"↑"のあるタイミングでキースキャンが行われた場合のプログラムの流れ（はじめてキーが押された場合）

このタイミングでキースキャンが行われた場合のプログラムの流れ（キーが離された時）

(6) 使用例

```
8200       XRA   A            AF
           LXI   SP,83C6H     31 C6 83
           STA   83ECH        32 EC 83
           PUSH  PSW          F5
           CALL  RGDSP        CD A1 01
           CALL  WAIT         CD 1C 82
           POP   PSW          F1
           INR   A            3C
           PUSH  PSW          F5
           CALL  INPUT        CD 23 02
           INR   A            3C
           JNZ   8228H        C2 28 82
           POP   PSW          F1
           JMP   8201H        C3 01 82
821C WAIT  LXI   D,0FFFFH     11 FF FF
           DCR   E            1D
           JNZ   $-1          C2 1F 82
           DCR   D            15
           JNZ   $-5          C2 1F 82
           RET                C9
8228       CALL  INPUT        CD 23 02
           JMP   8200H        C3 00 82
```

この例は，データレジスタの下位2桁にカウンタを構成しています．この時，キーボードのいずれかのキー（RESETは除く）が押された場合カウントを停止し，再びはなされた時にカウンタをリセットした後，カウントアップを始めるというプログラムです．

カウント動作時にこのキー入力サブルーチン(1)をコールして，キーが押されたかどうかをモニタしています．また，キーが押されると8228H番地にジャンプし，ここのキー入力サブルーチン(1)内でそのキーがはなされるまで待ち合わせて最初にもどります．

4．3．4 キー入力サブルーチン(2)

(1) サブルーチン名

KEYIN

(2) スタート番地

0216番地

(3) 入出力条件

入力パラメータ　　なし

出力パラメータ　　Acc＝入力データ
〔HEXA〕
キーフラグ セット

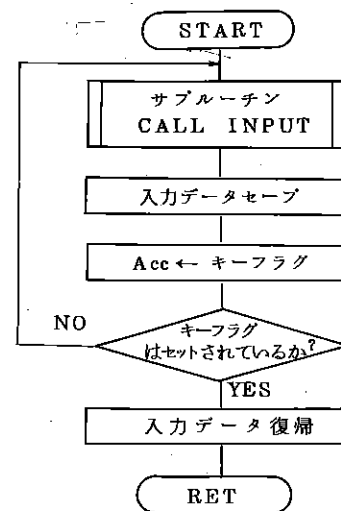
使用レジスタ　　A，F，B，D，E

使用スタック　　2レベル

(4) 機　能

このキー入力サブルーチンにおいては，キーボードスイッチがはじめて押されたことが検出されるまで，このルーチンの中でキースキャンを繰り返します．

キーボード・スイッチが押されたことが検出されたら抜け出ます．このときの処理は，4.3.3 キー入力サブルーチン(1)と全く同じです．

(5) フローチャート

4．3．5　シリアル出力サブルーチン

(1) サブルーチン名

SRIOT

(2) スタート番地

027C番地

(3) 入出力条件

入力パラメータ　　Acc＝出力データ

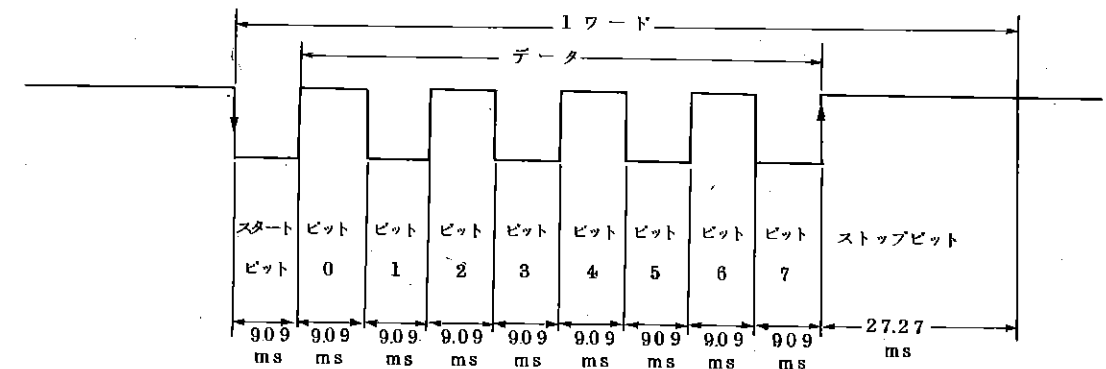
出力パラメータ　　なし

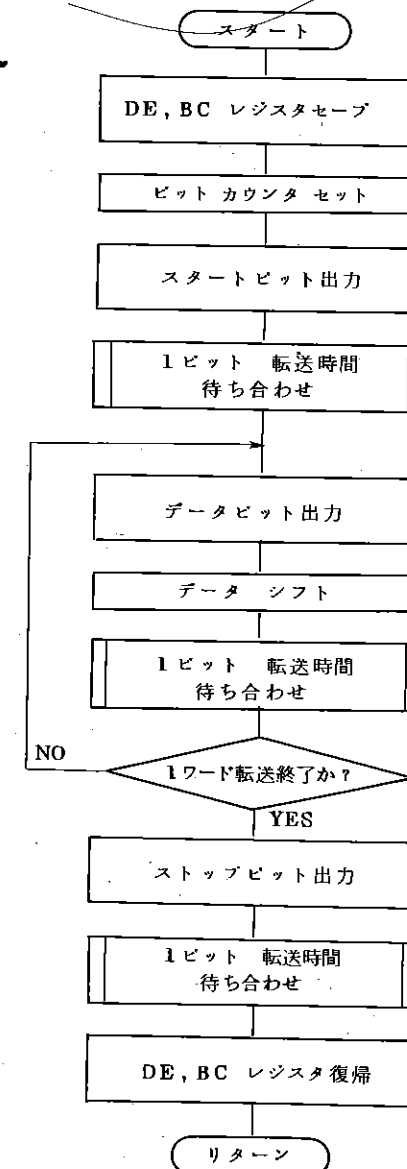
使用レジスタ　　A

使用スタック　　3レベル

(4) 機　能

Accに格納されている8ビットのデータを次のフォーマットに従ってシリアルデータに変換して，PPI（μPD8255）のポートCの，PC 0 端子（14番ピン）に出力します．スピードはテレタイプの転送スピードの110ボードに合わせてあります．

シリアル転送フォーマット

上記の例は“55”というデータをシリアルに変換したものです（最下位のビットから順に送られます）．

(5) フローチャート

(6) 使用例

```
        MVI    B, 0        8200   0600
        LXI    H, 8000H    8202   210080
    ┌→MOV    A, M        8205   7E
    │   CALL   SRIOT       8206   CD7C02
    │   INX    H           8209   23
    │   DCR    B           820A   05
    └─JNZ    8205H       820B   C20582
        HLT                820E   76
```

8000~80FF番地までの256バイトの内容をシリアルに転送します.

4.3.6 シリアル入力サブルーチン

(1) サブルーチン名

SRIIN

(2) スタート番地

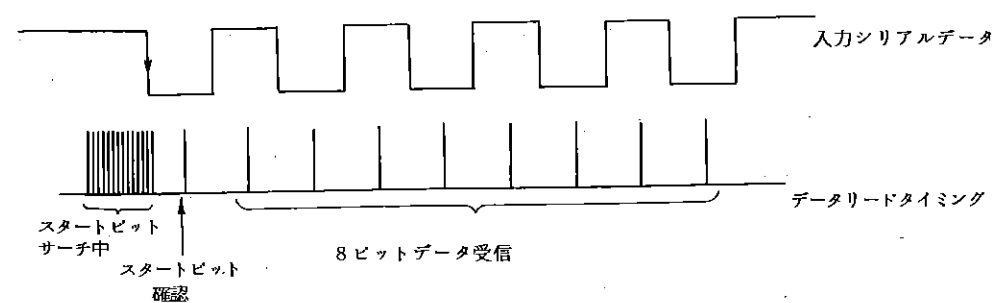
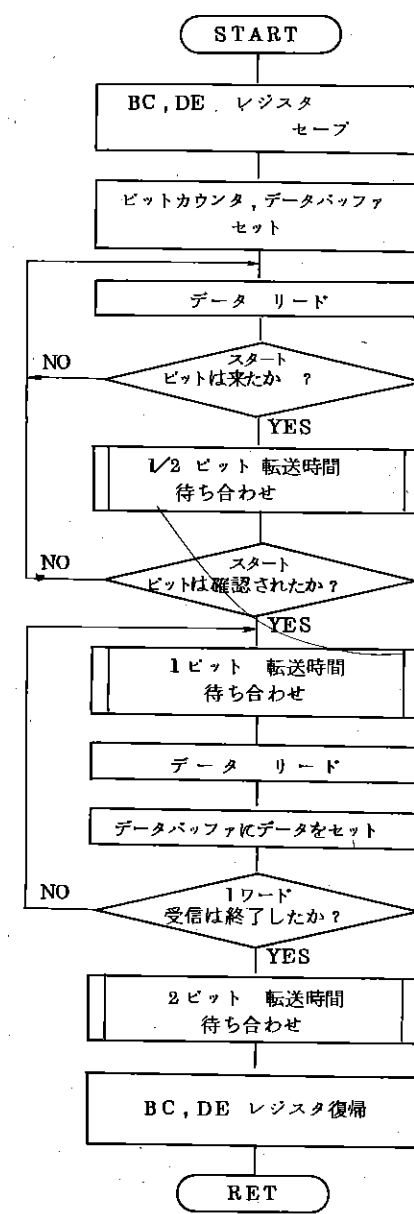
02A0番地

(3) 入出力条件

入力パラメータ	なし
出力パラメータ	Acc＝入力データ（8ビット）
使用レジスタ	A
使用スタック	3レベル

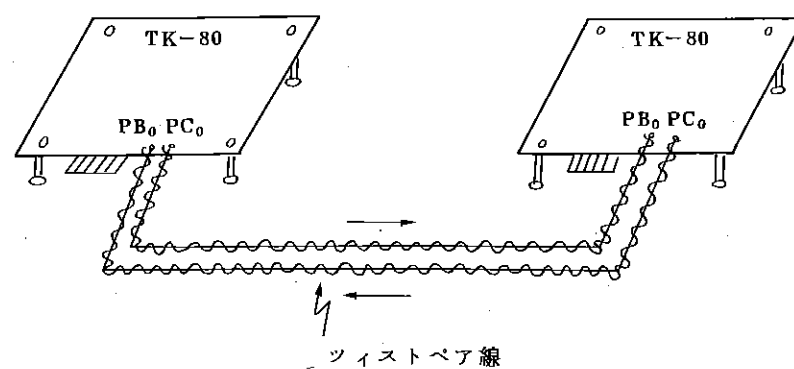
(4) 機　能

PPI（μPD8255）ポートB PB_0（18番ピン）に入力されるシリアルデータを受信して8ビットのデータに編集してアキュムレータにロードします.

PPIに入力されるシリアルデータは，スペースが"LOW"，マークが"HIGH"というレベルで入力します．サブルーチンでは，スタートビット（LOW）が来るまで待ち続け，スタートビットを受信すると，サブルーチン内のタイマプログラムによりインターバルタイムをカウントして，以降の8ビットのデータを受信します.

(5) フローチャート

(6) 使用例

連続して入ってくるデータをメモリの8100番地から次々と格納していきますが，"FF"というデータが入るとストップします.

```
LXI    H, 8100H      8000   210081
MVI    B, 0FFH       8003   06FF
CALL   SRIIN         8005   CDA002
MOV    M,A           8008   77
INX    H             8009   23
CMP    B             800A   B8
JNZ    $-6           800B   C20580
HLT                  800E   76
```

(7) 応用例

シリアル入力とシリアル出力のサブルーチンを用いて，2台のTK-80の間でデータの交換ができます．

ラインドライバ／レシーバを追加すれば，2台の距離は大きくできます．

4.3.7 タイマ・サブルーチン

(1) サブルーチン名

D1 : 4.5112 msec タイマ

D2 : 9.0176 msec タイマ

D3 : 27.0176 msec タイマ

(2) スタート番地

D1 : 02DD番地

D2 : 02EA番地

D3 : 02EF番地

(3) 入出力条件

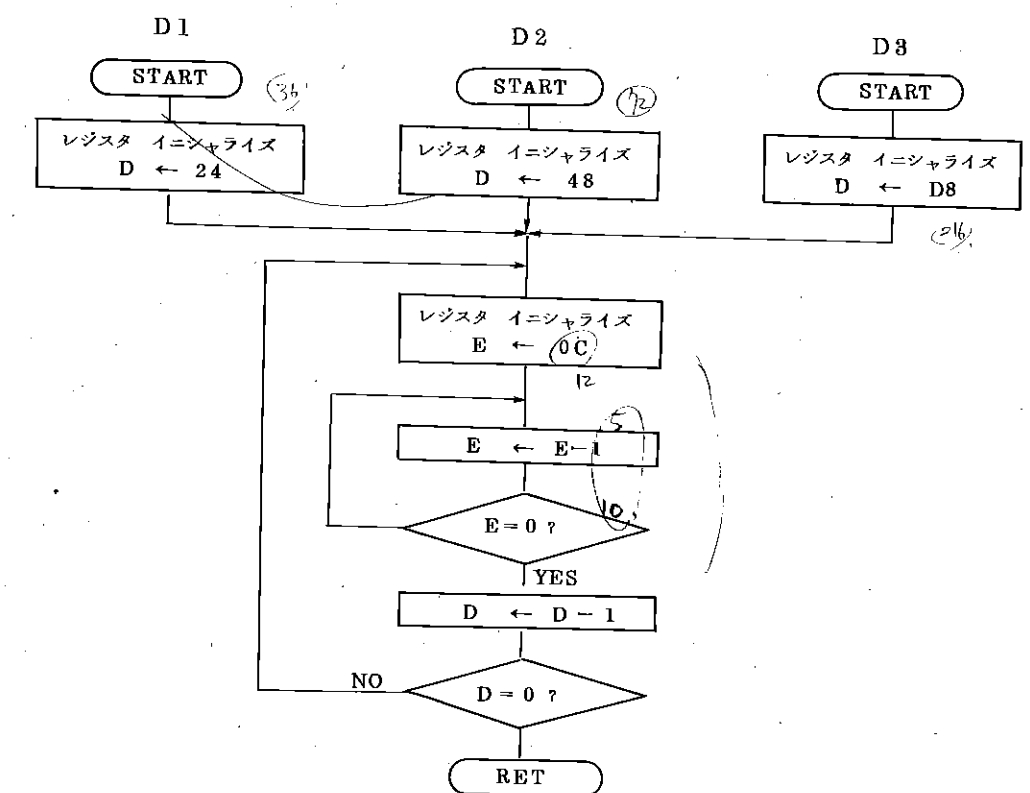
入力条件	なし
出力条件	なし
使用レジスタ	D，E
使用スタック	0レベル

(4) 機　能

ある処理とある処理との間に，これらのサブルーチンをコールすることにより，それらの処理間にコールしたサブルーチンに対応したインターバル・タイムをとることができます．

このサブルーチンは，イニシャライズされるレジスタの値により回数が決定されるループであり，そのループを通過するまでにCPUが費やす処理時間が，そのタイマ設定時間となります．

3つのサブルーチンは，シリアル入出力ルーチン（カセットルーチン）において，ビット間のインターバル・タイマとして使用しているため，その設定時間をD1は1/2ビット，D2は1ビット，D3は3ビットの転送間隔になるようにレジスタをイニシャライズしています．

(5) フローチャート

第5章　TK－80ハードウェア

本章では，TK－80がどのように設計され，ハードウェアとソフトウェアがどのように構成されているかを説明します．

基本的にはTK－80の設計に関して述べていますが，大部分が今後のシステム設計にも応用できるよう説明されていますので，本章を理解された方は，次にあなたのシステムを自分で設計することができるようになっていることでしょう．

5.1　マイクロコンピュータの基本的なシステム構成

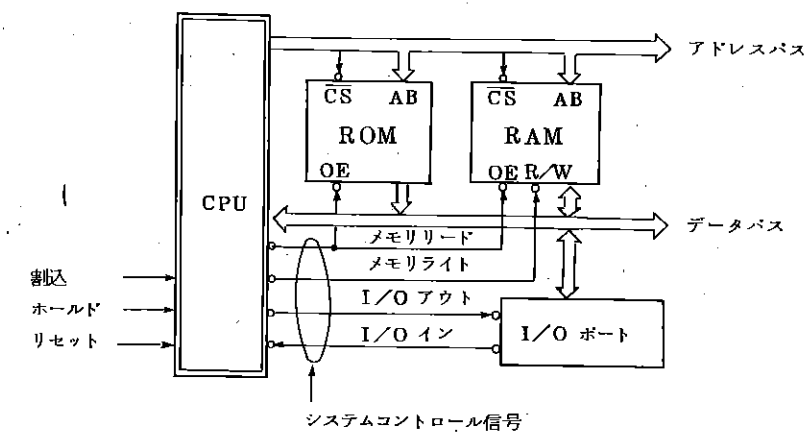
図5－1　マイクロコンピュータの基本的なシステム構成

マイクロコンピュータの基本的な構成は，図5－1のようになります．基本的にはCPUの部分とROM，RAM，I/Oポートで最小のシステムが構成されます．構成要素の具体的な説明は，5.2で述べますが，ここでは各要素の役割について大まかにとらえてください．

図5－1でメモリをROMとRAMに区別して書いたのは，一般的にマイクロコンピュータがなんらかの装置に組み込まれる場合，制御プログラムはROMに固定化して書き込まれることが多いからです．

ROMに制御プログラムを書き込んでおけば，電源を入れるとすぐプログラムを実行できる状態にあるわけですから，わざわざプログラムを外部から読み込ませる必要もなくなり，高価な入力装置を備える必要もなくなります．このあたりがミニコンピュータとの考え方の大きな違いと言えます．このようにマイクロコンピュータではプログラムをROMに固定化することによって，専用化された使われ方をされることが多いわけですが，ROMを差し換えれば異なった機能のシステムに生れ変わることもできます．RAMにはプログラムで処理するためのデータを格納したり，プログラムそのものを書いたりします．さらにμPD8080Aでは，スタックをRAMで構成されるメモリ領域の中に確保しています．

プログラムは，ROMに書かなければいけないということはなく，あくまでもROM化した方が経済的であり，かつ信頼性もあげられるという場合に使われるわけです.

I／Oポートは，コンピュータの内部と外の世界とのデータ交信を行う部分であって，基本的にはパラレルI／OポートとシリアルI／Oポートがあります．パラレルI／Oポートというのは，コンピュータ内部のデータバス上の信号を並列に外部に出力したり，または内部バスに取り込むためのユニットであり，その主たる機能は必要な時刻にタイミングを合わせてデータをラッチ（つかまえて保持する）したり，必要なユニットのデータだけをバスに読み込むというようなものです.

これに対してシリアルI／Oポートは，外部とのデータ交信を直列データで処理するもので，CPU側の並列データと調整するために並列－直列，直列－並列変換回路を内蔵しているのが普通です.

例えば，μPD8255はパラレルI／Oポート，μPD8251はシリアルI／Oポートとしての機能を備えています.

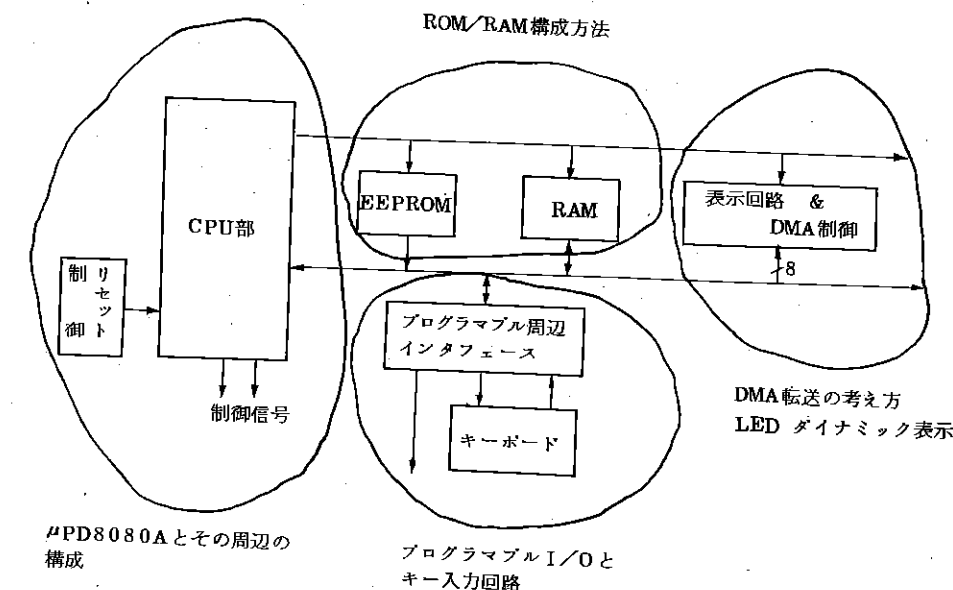
CPU部分は，ROMまたはRAMに書かれたプログラムを遂次実行して，データの処理を行ったり，I／Oポートとデータのやりとり等の制御を行ったりします.

ROM，RAM，I／Oポート等は，それ自身ではデータを加工する能力はありません．データを加工したり，プログラムを命令と理解して処理実行するのはCPUの任務です．CPUはメモリをアクセスするために番地を指定するアドレスバスとデータをやりとりするデータバス，およびデータの送受信されるべきタイミングを外部に指示するコントロール信号，外部から直接CPUの動作を制御するための制御端子（割り込み，ホールド，リセット端子等）をもっています．以上が大体マイクロコンピュータを構成する基本的な要素です.

5.2 TK－80のシステム構成

それでは，具体的にTK－80がどのように構成されているかを，各部について詳細に説明します．その前にTK－80のシステムブロック図を図5－2に示しておきますので，全体の概要を頭に入れておいてください.

図5－2　TK－80システムブロック図

ROM／RAM構成方法
CPU部
EEPROM
RAM
表示回路 & DMA制御
8
制御
リセット
プログラマブル周辺インタフェース
キーボード
制御信号
DMA転送の考え方
LED ダイナミック表示
μPD8080Aとその周辺の構成
プログラマブルI／Oとキー入力回路

CPU部はCPUチップ（μPD8080A）とその他若干のICで構成されています．PROMにはモニタプログラムが書かれており，TK－80の基本的な動作は，このプログラムで実現されているわけです．CPU部にリセットがかかると（RESET キーを押す），このモニタプログラムが走りはじめます．このモニタプログラムは，プログラマブル周辺インタフェース（以下PPIと略します）を介して，キーボード・スイッチ25個を常時スキャンしながら，どのキーが押されたかを調べています.

RAMは256ワード×4ビットのICが2個で，最小の256バイトが構成されます．ボード上には8個まで実装できます.

表示回路は，8個の7セグメントLED（発光ダイオード）で構成されており，基本的にはメモリアドレスとデータを16進数で4桁ずつ表示します．表示のためのデータは，DMA転送により行っていますので，表示データ転送のためのプログラムを書く必要はありません.

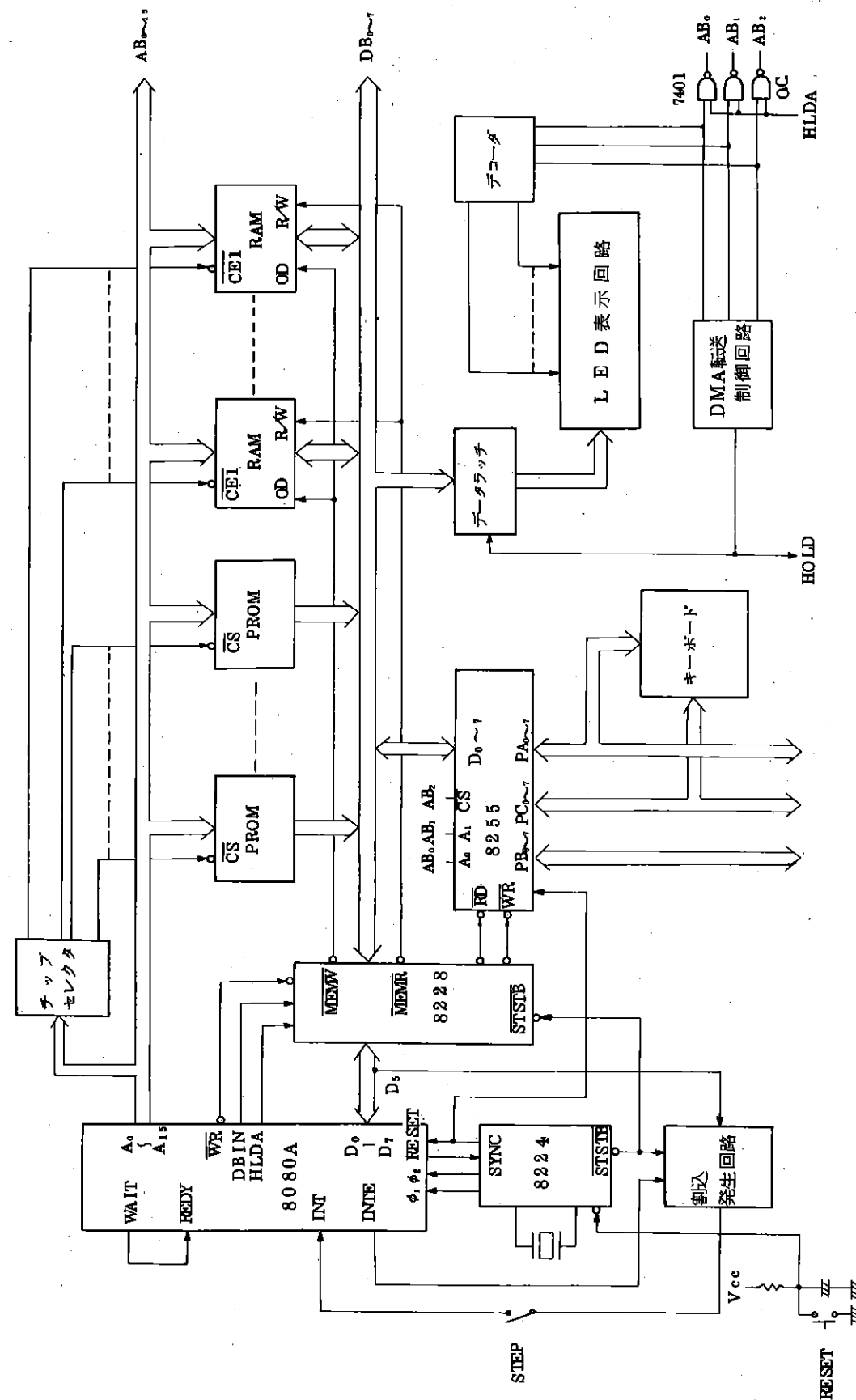
図5－3　TK－80のシステム構成

5・2・1　CPU部のシステム構成

図5－4　CPU部の基本構成

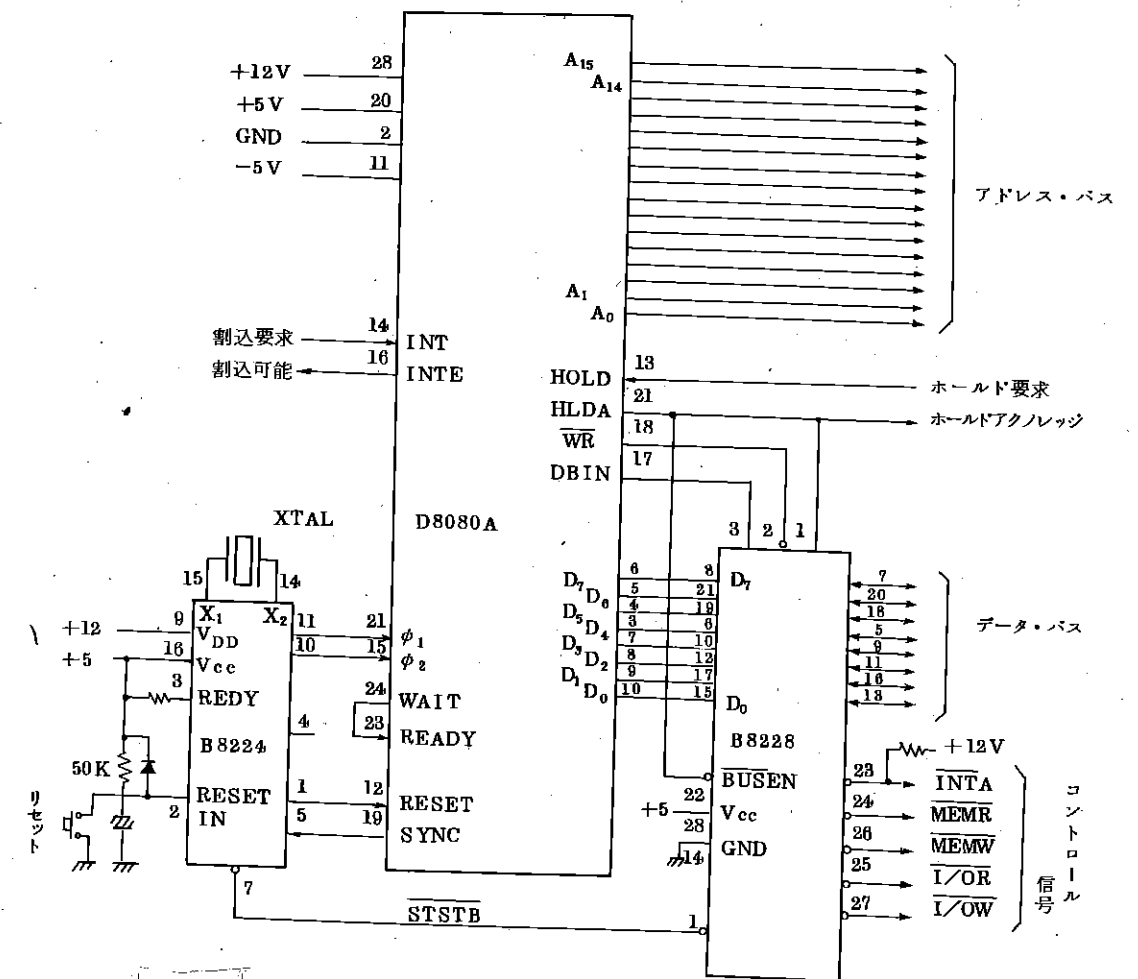
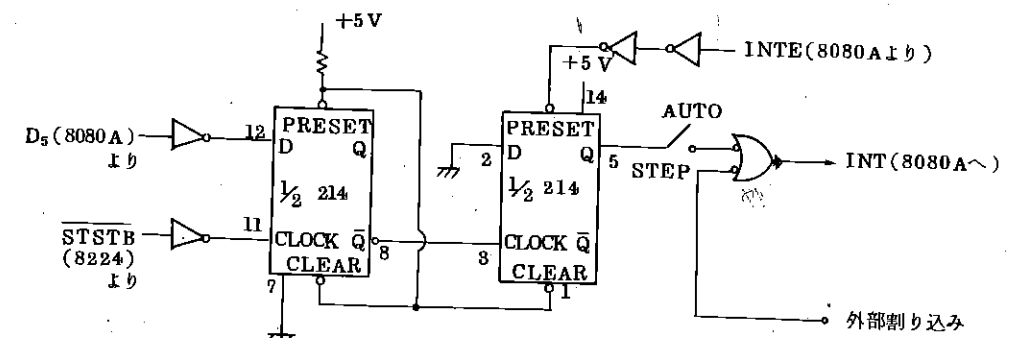



図5－5　1命令ステップ割り込み信号発生回路

CPU部の基本構成を図5－4に示します．CPU（μPD8080A）にクロック・ジェネレータ（μPB8224），システム・コントローラ（μPB8228）を接続するだけの簡単な回路ですみます．CPUは電源として＋12V，＋5V，－5Vが必要ですが，TK－80では便宜上，－5V電源をμPB8224のOSC出力端子の発振信号を検波することにより，約－5Vの電圧を得ており，ボードの外部からは＋12V，＋5Vの2種類を供給するだけですみます．μPB8224のリセット端子には，電源を投入したときに自動的にリセットを行う回路と，手動のリセットスイッチがつけられます．クロックの発振は，CPUのクロックの9倍の基本発振周波数をもつ水晶振動をμPB8224のX_1，X_2端子に直接接続すれば得られます．

μPD8080AのWAIT出力端子とREADY入力端子とを直結しておけば，2MHzのクロックで動作させてもアクセスごとに1ウェイトとられますので，アクセスタイムの遅いメモリ（例えばCMOS RAM μPD5101C－E等）でも使用できます．この方法では，メモリがアクセスされていないサイクルでも，余分な待ち時間がとられてしまいますが，複雑な回路が一切必要ないという理由で採用してあります．

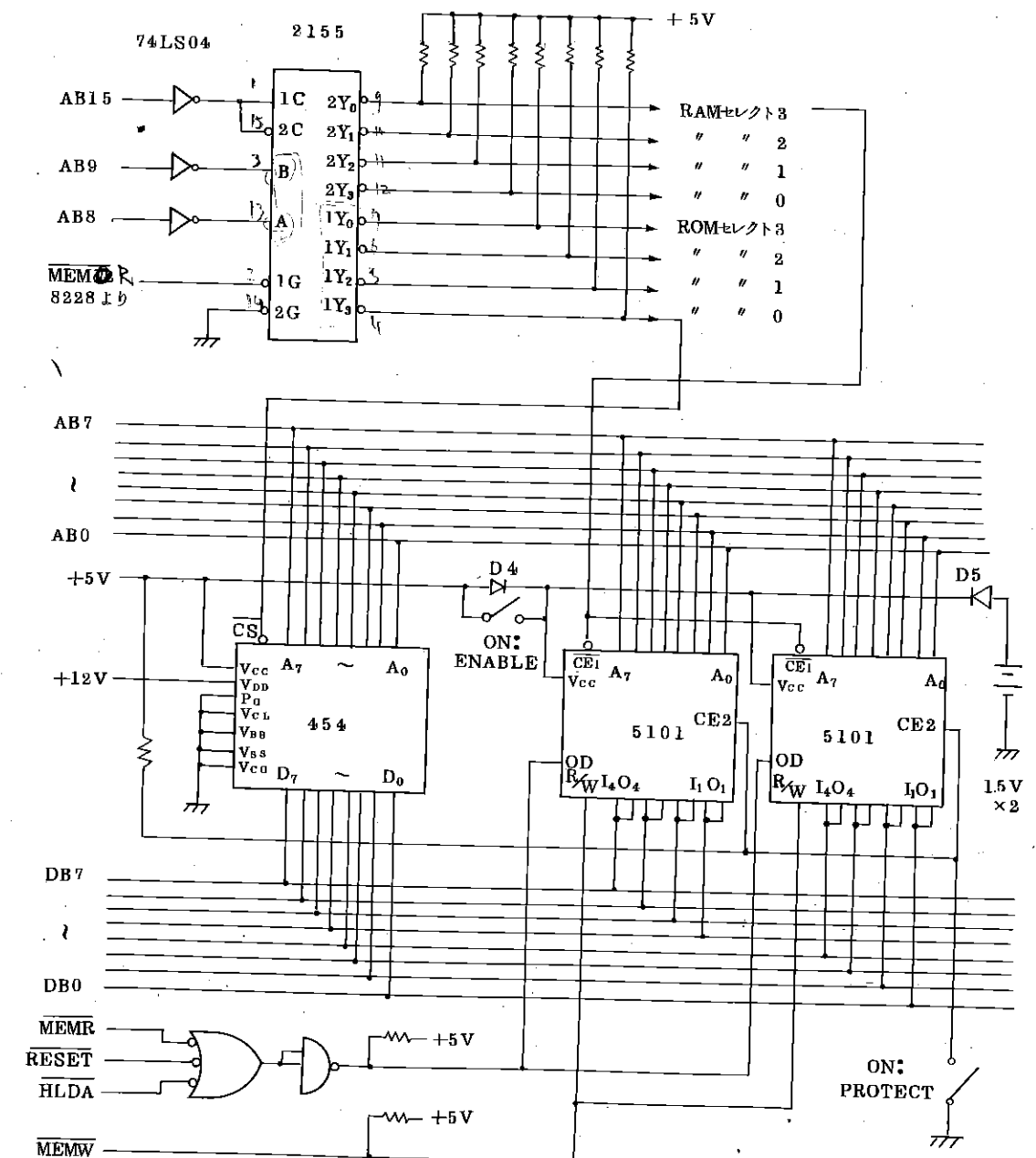
INT入力端子には割り込み要求信号が接続されます（通常ロウレベルで，要求時にハイレベルとなる信号です．…アクティブ・ハイ信号という表現をします）．INTE出力端子には，CPU内のインタラプト・イネーブル（割り込み可能）F／Fの状態が出力されます．

HOLD入力端子には，DMA転送を行いたい時刻にホールド要求信号を入力します（アクティブ・ハイ信号）．ホールド要求が受け付けられるとHLDA出力がハイレベルになります．外部回路でこの信号に同期させてメモリをアクセスすれば，CPUの動作とDMAが競合することを避けられます．

μPB8228から出るコントロール信号は5種類あります．$\overline{\text{INTA}}$端子は割り込みレベルが1つの時には抵抗を介して＋12Vにプルアップしておきます．$\overline{\text{MEMR}}$，$\overline{\text{MEMW}}$は，各々メモリのリード，ライト時に発生されます（アクティブ，ロウ信号）．$\overline{\text{I/OR}}$，$\overline{\text{I/OW}}$は入力デバイスからのリード，およびライト時に発生されます（アクティブ，ロウ信号）．アドレスバスはμPD8080Aから直接取り出して使用していますので，DC的な負荷は標準TTL 1個しか取り出せません．TK－80では，アドレスバスに接続されるメモリはすべてMOSタイプであり，デコーダもローパワー・ショットキーTTL（低レベル入力電流が360μA以下）を使用していますので，DC的にはこれで問題はありません．交流的には1つのバスに接続されるMOSゲート（入力容量は数pF～10pF）の個数が増えれば，アドレス信号の立ち上り，立ち下り波形が鈍り，アクセス時間に影響してきます．TK－80では，メモリアクセスの時間を充分とってありますので，この点は余裕のあるよう設計されています．データバスは，μPB8228の内部で双方向性ドライバを通ってドライブ能力が強化されていますので，標準TTLを6個までは接続できます．μPB8228の$\overline{\text{BUSEN}}$端子にHLDA信号を加えているのは，バスドライバをハイ・インピーダンスの状態にして，DMA転送のデータとかち合わないようにするためです．基本的にはこの回路でCPU部は完成しますが，TK－80ではプログラムのデバグの便宜のため，1命令ごとの割り込み要求信号発生回路を追加してあります（図5－5）．原理的にはμPD8080AからINTE（割り込み可）信号が出たら，次の命令を実行した後すぐ割り込み信号が発生されるような回路です．μPD8080Aへの割り込み信号としては，このステップ実行割り込み信号と外部割り込み信号の論理和をとった信号を用います．

5・2・2　ROM，RAMの構成

TK－80のボード上には，ROM，RAMとも最大1.024バイトずつ実装できます．ROMは，μPD454（256ワード×8ビット）を4個，RAMはμPD5101（256ワード×4ビット）を8個取り付けることになります．図5－6にメモリ最小構成を示します．

図5－6　ROM／RAM最小構成

454，5101とも256ワードのメモリですので，アドレス入力端子は8本あり，それぞれアドレスはAB_0～AB_7へ接続されます．さらに256ワード単位でチップの選択を行う必要があります．これは2155(DUAL 2 TO 4 DECODER)で行っています．2155の入力端子に74LS04が入っているのは，アドレスバスの負荷を少なくするためのものです．TK−80ではROMかRAMをAB_{15}（アドレスの最高位ビット）で切り変えていますので，8000番地以上がRAMエリアとみなされます．2155の出力信号は，チップセレクト信号としてROM，RAMにつながれています．各出力端子にすべてプルアップ抵抗がついているのは，MOS ICのハイレベル入力電圧の最低値である3.0Vを確実にするためです．5101のOD（出力ディスエイブル）端子には，$\overline{\text{MEMR}}$，$\overline{\text{RESET}}$，$\overline{\text{HLDA}}$の論理和信号を加えていますので，これ以外のタイミングでは，5101の出力はハイ・インピーダンス状態となります．R/W端子には，8228からの$\overline{\text{MEMW}}$信号が直接接続されていますが，ここにもやはりプルアップ抵抗がつけてあります．このプルアップ抵抗は8228の$\overline{\text{BUSEN}}$入力信号がハイのとき，制御信号が不確定（ハイ・インピーダンス）になるのを避けて，メモリをリード状態に保つためです．

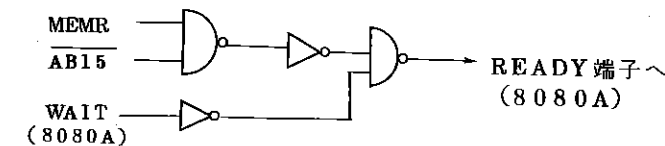
CE_2端子は通常動作時には，Vcc へプルアップしてありますが，トグルSWをONにするとGND電位となり，メモリのリード／ライトの両動作ともプロテクトされます．このSWをONにした後＋5Vの電源をOFFにしても，バッテリー（例・単三乾電池2本）の電圧が印加されている限り，データは保存されます．

ダイオードD1に並列に入っているSWは，メモリプロテクトSWと連動しており，メモリが動作中はD1を短絡し，ダイオードによる電圧降下をさけるためのものです．

TK−80では，RAMは上位アドレスのチップから実装しています．なぜならモニタプログラムのワーキングエリアとして，最上位のアドレスを使用しているからです．

（実装順位）	（番　地）	（セレクト信号）	（RAM番号）
①	8300～83FF	セレクト3	4，8
②	8200～82FF	〃　2	3，7
③	8100～81FF	〃　1	2，6
④	8000～80FF	〃　0	1，5

TK−80の基本キットには，RAMとしてCMOSタイプの5101を使用していますが，これと全くピンコンパチブルな2101がNMOSタイプで揃っています．バッテリ・バックアップの必要のない方または電池の消耗を問題としない方は，2101がそのまま使用できます．

5101と2101を組み合わせて実装することもさしつかえありません．454，5101/2101ともに完全なスタティック動作をしますので，動作は安定しておりVccラインにのるノイズはそんなに大きなものではありませんので，バイパスキャパシタにもそれ程厳しい特性は要求されません．TK−80では普通のセラミックキャパシタを使っています．RAMに2101だけを使用する場合には，アクセスタイムが十分速いので，RAMがセレクトされるときCPUを待たせる（WAIT）必要はありません．このときには，PROMがアクセスされたときだけWAITをかければ，CPUのむだな待ち時間が節約できます．

（0～7FFF番地がアクセスされた時のみワン・ウェイトとなる回路）

5．2．3　表示回路とDMA転送

図5－7　表示回路

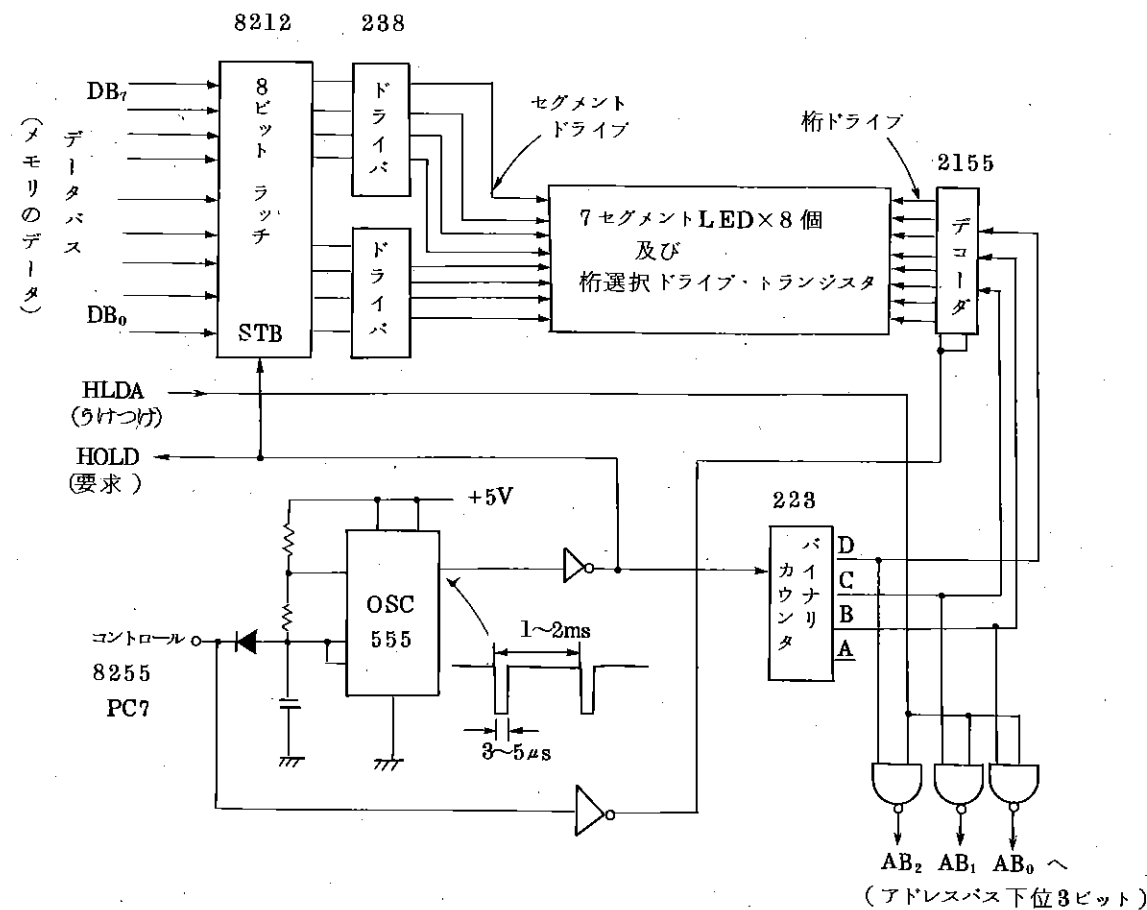
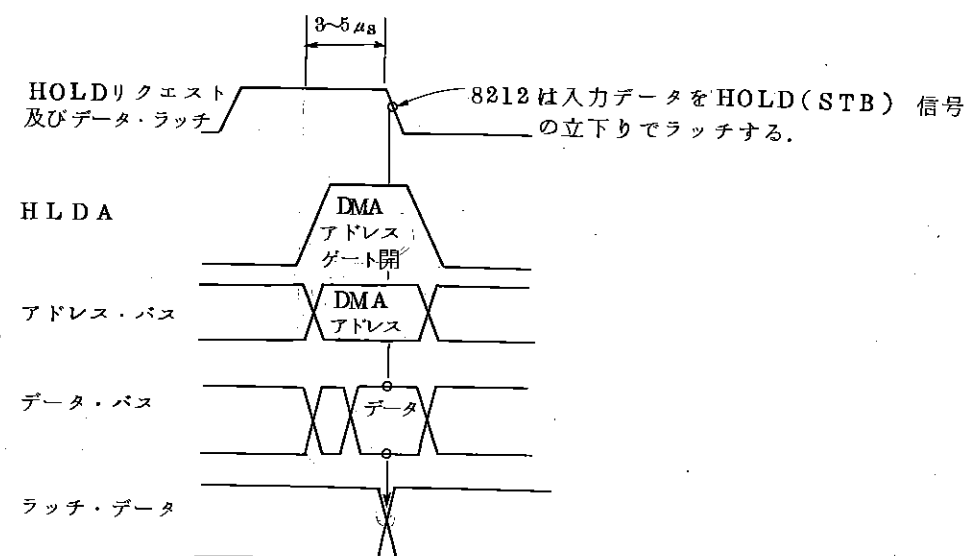



図5－8　DMA転送のタイミング

表示回路は図5－6に示すように，DMA転送回路と8桁LEDのダイナミック表示回路を組み合わせたものです．前にモニタプログラムで述べたように，LEDの各セグメントに対応する表示データは，RAMの最後の8番地（83FF～83F8）に入っていますので，これを直接読み出して表示しようとするものです．この回路では，まずOSC（555タイマ用ICでつくってある）は，1～2ms間隔で数μsのパルスをHOLD要求信号としてCPUへ送ります．このパルスはまたバイナリカウンタ（223）のカウントパルスとしても使われており，1回ごとにアクセスするメモリの番地を進めていきます．HOLD要求が受け付けられると，HLDA信号が送り返されます．HLDA信号がきたときには，8080Aのアドレスバスおよびデータバスはハイ・インピーダンスになっていることを示していますので，この信号でゲートを開いてアドレスバスの下位3ビットへDMAアドレスを接続します．残りの上位ビットは，プルアップ抵抗でつってありますので，DMAアドレスが定まると，RAMのアクセス時間だけ遅れてデータが読み出されてデータバス上にあらわれます．このデータは，HOLD要求信号の立ち下りのタイミングで8ビットのラッチにセットされ，次のデータが再び読み出されるまで1つの桁の表示に使われます．この動作が1～2msおきに繰り返し行われ，順次アドレスがスキャンされそれに応じて表示桁も移動していくわけです．HLDAが出ている間は，CPUはプログラムの実行を一時中断しているわけですが，全体の時間に比べると数μsというのは極めて短い時間ですので，実用上はまずさしつかえないでしょう．しかし正確にタイミングを計算したいようなプログラムでは，ときとして問題となる場合があります．この場合には555の発振回路を強制的に止めてやれば，HOLDの要求は起こりません．

この制御は，8255のPC₇端子をロウレベルにすることで実現しています．発振が停止するとLEDのスキャンニングも止まり，1つの桁だけを点灯することになり，熱的設計を満足しませんので，このときには桁ドライブ回路をディスエイブル状態にするようにしています．

発振の周期を長くすればそれだけCPUが止まる割合を小さくできますが，ダイナミック点灯の周期は少なくとも50サイクル以上にしないと人間の目にはチラついて見えます．8桁分ありますので，1桁分は2.5ms（2.5×8=20ms）程度が限度と考えてください．

TK－80では，1～2msの間に入るようにしてあります．

5．2．4　プログラマ・ペリフェラル・インタフェース(PPI)とキーボード回路

コンピュータが外部からデータを読んだり，逆にCPUから外部にデータを送り出す場合に使われるのがペリフェラル・インタフェースです．

TK－80では，μPD8255(PPI)を使用しており，キーボードのスキャンニング回路と外部インタフェースを1個のLSIで構成しています．

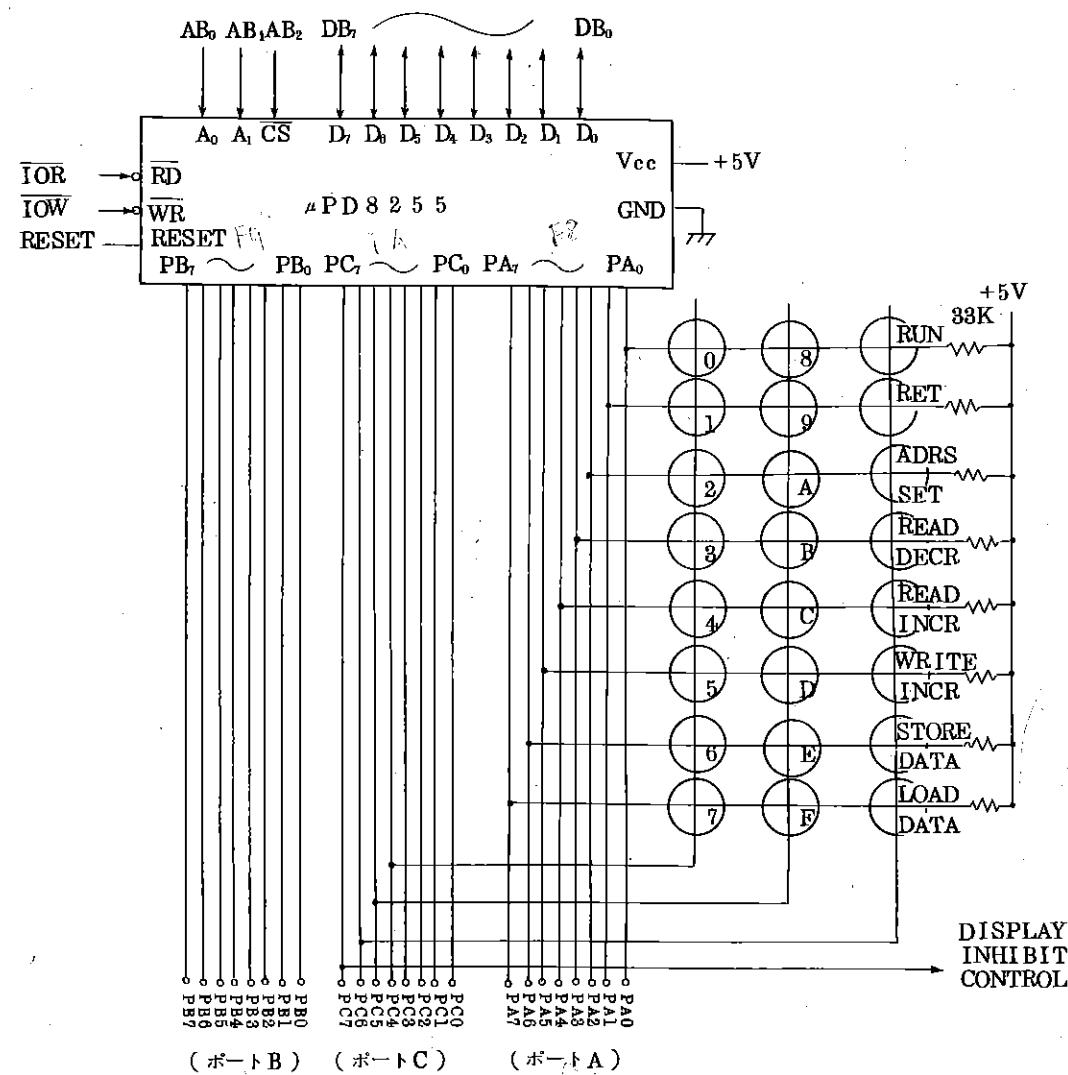
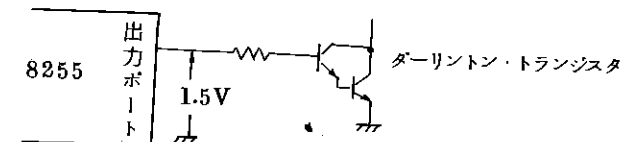
図５－９　μPD８２５５とキーボード回路

各ポートは入力にも出力にもなります．

図５－７に８２５５を使用したペリフェラル・インタフェースとキーボード回路を示します．８２５５は８ビットの入出力ポートを３つ備えており，それぞれポートA，B，Cと呼ばれます．各ポートはプログラムで入力ポートにも出力ポートにもすることができ，それぞれが８ビットデータの送受信をすることができます．

ポートCだけはさらに特殊な機能をもち，８ビットのデータのうち任意のビットを指定して，セットしたりリセットしたりすることができます（ビットセット／リセット機能）．８２５５の詳細な機能は別紙開発速報を参考にしていただくとして，ここではキー回路の動作と８２５５の関係を述べておきます．キーボードをスキャンするためにポートCは出力に，ポートAは入力にプログラムされています．モードは０です．ポートAの全端子は抵抗でプルアップしていますので，キーが押されない限りポートAから読み込まれるデータはすべてハイレベルです．PC_4，PC_5，PC_6は，キー共通ラインをロウベルで順番にスキャンするようモニタプログラムが作られていますので，ど

の列のキーが押されたかはモニタ自身で判断できるわけです．スキャンプログラムの詳細は，モニタプログラムの説明の章で述べてあります．

次に外部とのインタフェースの電気的な性能について説明します．

各端子が出力にプログラムされたとき，標準TTL１個のドライブ能力があります．さらに出力ポートに直接ダーリントン・トランジスタを接続してドライブすることも可能です（ただし最大消費電力の問題で８端子以内に制限されます）．

このような使い方が８端子まで可能

５．２．５　μPD8255のプログラミング法

８２５５はプログラマブル・インタフェースであって，３つのポートプログラムによってモード０，１，２のいずれかを指定して使うことができます．モード１，２は外部機器とのデータ転送を割り込みを用いたハンド・シェイキングという方法で実現できる高度な使い方ですので，このことに関しては個別開発速報を参考にしてください．

ここでは最も基本的に使われるモード０での使い方を説明しておきます．

図５－10　モード０でのμPD８２５５の各ポートの機能

○ポートA（８ビット）

入力ポート／出力ポートの指定ができます．

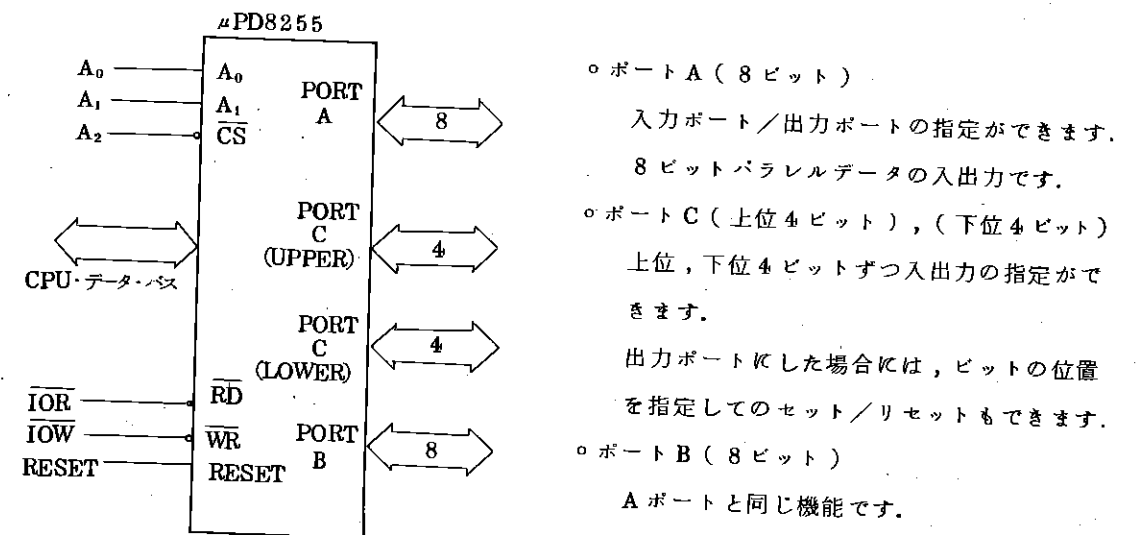
８ビットパラレルデータの入出力です．

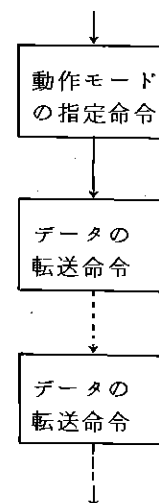
○ポートC（上位４ビット），（下位４ビット）

上位，下位４ビットずつ入出力の指定ができます．

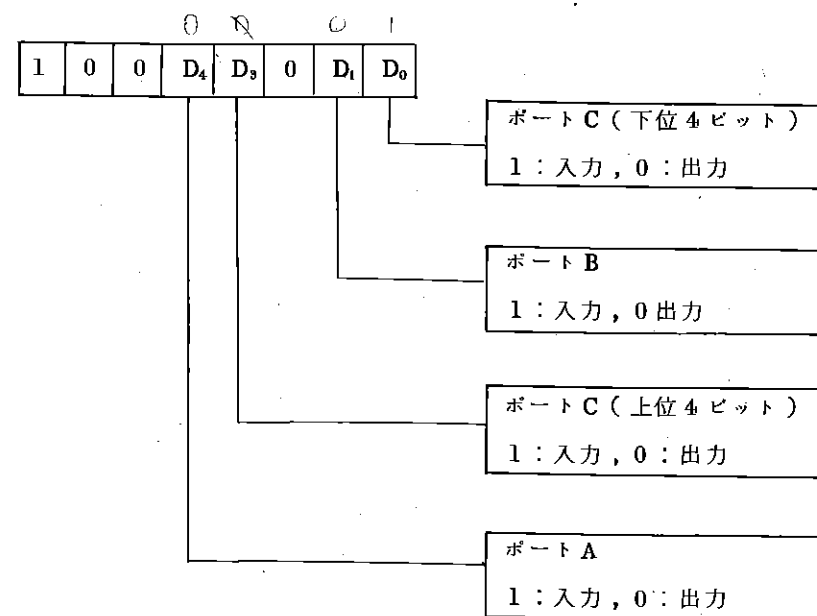
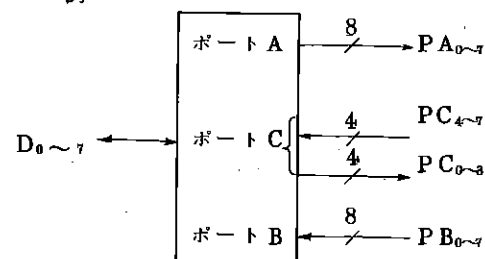
出力ポートにした場合には，ビットの位置を指定してのセット／リセットもできます．

○ポートB（８ビット）

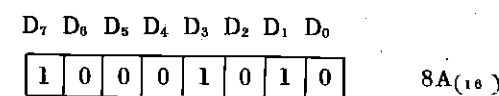
Aポートと同じ機能です．

８２５５はRESET信号を受けると，すべてのポートが入力モードに戻されて，以後命令で新しいモードがセットされない限りこの入力モードは継続されます．

８２５５のプログラムによる動作は次のようになります．

(1) 動作モードの指定

動作モードは，コントロール・ワードと呼ばれるデータを８２５５へＯＵＴ命令で送り込むことによりセットされます。

モード０でのコントロール・ワードの各ビットの指定は次の通りです。

例

この例の入出力組合せとするためには，コントロールワードは

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	0

$8A_{(16)}$

のようになります。

このコントロール・ワードをデータとして，出力命令で８２５５へ送り込みます。

このときのプログラムは次の通りです。

コーディング	機械語	
MVI A,08AH	3E 8A	……コントロール・ワードの値をＡにロード
OUT 03	D303	……コントロール・ワードを８２５５へ送り込む。

（ＯＵＴ　０３の０３というＩ／ＯアドレスはＴＫ－８０での値です）

以上で動作モードのセットができます。

(2) データの出力

８２５５へデータを送り込むには，ＯＵＴ命令を使いますが，このときのＩ／Ｏアドレスはポートによって異なります。

コーディング	Accの内容	機械語
OUT 00	……………ポートＡ	D300
OUT 01	……………ポートＢ	D301
OUT 02	……………ポートＣ	D302

(3) データの入力

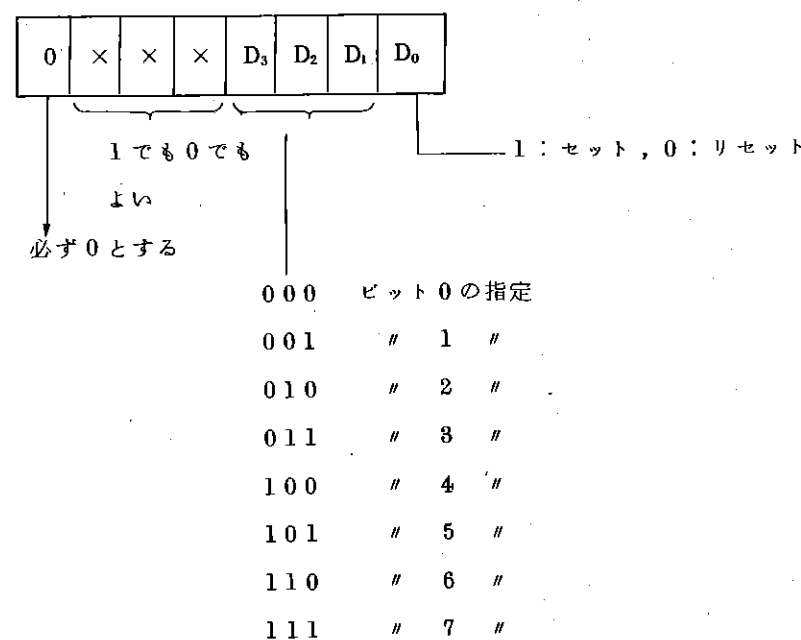
８２５５の各ポートからデータを入力する（ＣＰＵにAccにロードする）には，ＩＮ命令を使います。このときもポートによってアドレスが異なり，次のように対応しています。

IN 00	ポートＡから	DB00
IN 01	ポートＢから	DB01
IN 02	ポートＣから	DB02

例　ポートＡに01010101を出力し，ポートＣの上位４ビットを読み取る。

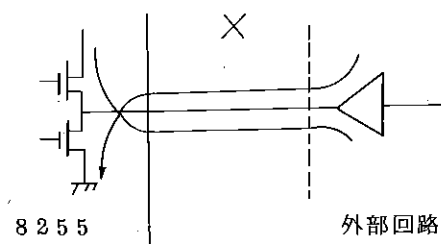
```
MVI  A,08AH  } モードのセット。これはプログラムの最初に一度行うだけで
OUT  03      } よい。
   ∫
MVI  A,055H  } 01010101→Acc
OUT  0       } (Acc)→ポートＡに出力
IN   02      } ポートＣのデータをAccに読み込みます。
```

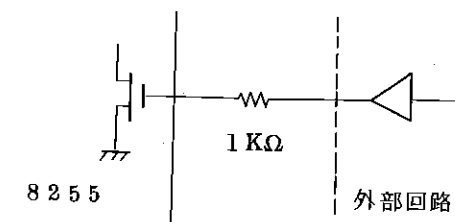
(4) ポートＣのビットセット／リセット

ポートＣが出力としてプログラムされているときは，そのビット位置を指定して，１命令でセットまたはリセットを行うことができます。これはコントロール・ワードを８２５５へ送り込むことで実現されます。

例　ポートCのビット2だけを1にして，その後再び0にする．他のビットの状態はそのままにしておきたい．このときのプログラムは次のようになります．

8255程のプログラマブルな能力が必要ないが，簡単にI／Oポートを実現したい場合は，μPB8212（8ビットI／Oポート）を使ってください．基本的な使用法は個別開発速報を参考にしてください．この場合も複数のポートの選択は，前述のようなアドレスの割り当て方法で行います．

5．2．6　μPD8255の使用上の注意事項

(1)　出力に指定したピンは，絶対外部からドライブしないでください．過大電流が流れます．

(2)　入力に指定したピンを誤って出力にプログラムしないでください．

ミスプログラムによる事故を防ぐために入力ピンには1KΩ程度の抵抗を直列につないでおくことをお勧めします．

(3)　8255はRESET信号により，3つのポートがすべて入力モードとなります．

(4)　8255のポート端子を外部へ引き出す場合には100KΩ程度のプルアップ抵抗をつけてください．

(5)　外部回路の電源がTK－80と別系統の場合は，先にTK－80の電源を入れ，リセットした後で外部回路の電源を投入するように心掛けてください．

(6)　外部回路から8255の端子へ過大電圧がかからないように注意してください．

5．2．7　アドレス／データ信号端子

TK－80のアドレスバスとデータバスは，プリントボードの端子に配線されています．

アドレスバス	ボード端子	データ・バス	ボード端子
AB 0	B 17	DB 0	B 33
〃 1	〃 16	〃 1	〃 32
〃 2	〃 15	〃 2	〃 31
〃 3	〃 14	〃 3	〃 30
〃 4	〃 13	〃 4	〃 29
〃 5	〃 12	〃 5	〃 28
〃 6	〃 11	〃 6	〃 27
〃 7	〃 10	〃 7	〃 26
〃 8	A 17		
〃 9	〃 16		
〃 10	〃 15		
〃 11	〃 14		
〃 12	〃 13		
〃 13	〃 12		
〃 14	〃 11		
〃 15	〃 10		

ボードの他の端子は空き端子となっていますので，必要な信号は適当な端子へもってくることができます．

第６章　TK−80CMTインタフェース

はじめに

ＴＫ−８０は，そのプログラムエリアとして0.5Ｋバイト（最大１Ｋバイト）のＲＡＭが実装されており，アプリケーション・プログラムを実行させるためには，まずキーボードより機械語で，プログラムを書き込んでおく必要があります．

ＴＫ−８０のＲＡＭは，Ｃ−ＭＯＳのデバイスを使用しているため乾電池を接続することにより，電源を切った後もＲＡＭ内のプログラムを保持しておくことができますが，それもその時ＲＡＭに書き込まれているプログラムのみで，いくつものプログラムを長時間保持しておくことは不可能です．

そこでこの章で述べる簡単なインタフェースをＴＫ−８０ボードのユニバーサル基板の部分に作ることによって，普通のカセット式ポータブル・テープレコーダをＴＫ−８０に接続し，テープにＲＡＭ内のプログラムを録音していくつものプログラムをファイルしておき，必要な時にＴＫ−８０のＲＡＭにテープにファイルされているプログラムをロードし実行させることが可能となります．

テープレコーダとのデータの送受は，モニタプログラムが管理しキーコマンドにより簡単に行わせることができます．

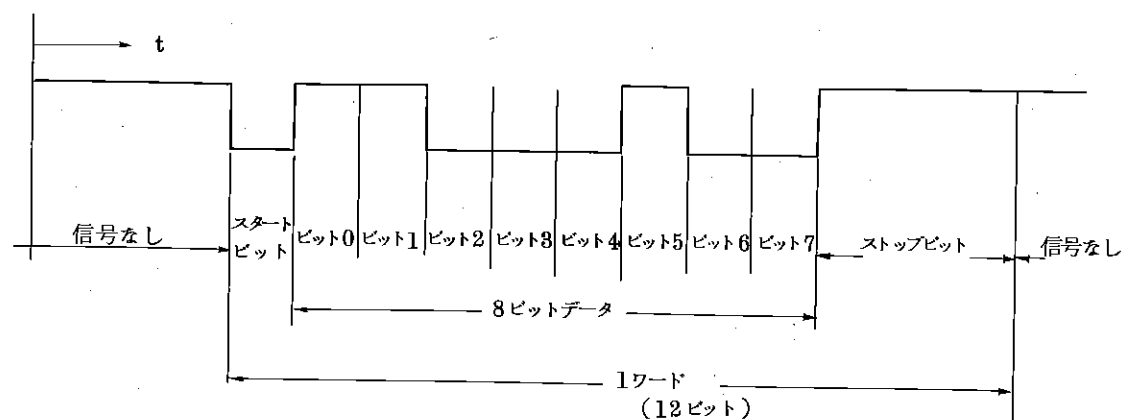
6.1 概　　要

このインタフェースは，デジタル信号を可聴帯域のオーディオ信号に変調して，テープレコーダに出力する変調回路，テープレコーダからのオーディオ信号を再びデジタル信号に復調してシステムに供給する復調回路とで構成されています。

本説明書では，転送データの形式，変復調原理，さらにインタフェースの製作方法について述べていきます。

6.2 データのフォーマット

データは，1ワード（8ビット）を1つの単位として，各ビットのデータをシリアルに転送します。1ワードの転送フォーマットは次のようになっています。

図6－1　データ・フォーマット

1ワードの転送用データは，スタートビットではじまりストップビットで終了します。

スタートビットは，これから1ワード分のデータを転送することを示す同期用ビットで，1ビット長のロウレベル信号を転送します。

スタートビットの直後から1ワード（12ビット）のデータを，下位ビットよりシリアルに転送していきます。この時の論理は，データ"1"はハイレベル，データ"0"は，ロウレベルとなります。

8ビットのデータの後，3ビット長のストップビットを転送して1ワードのデータは終了します。

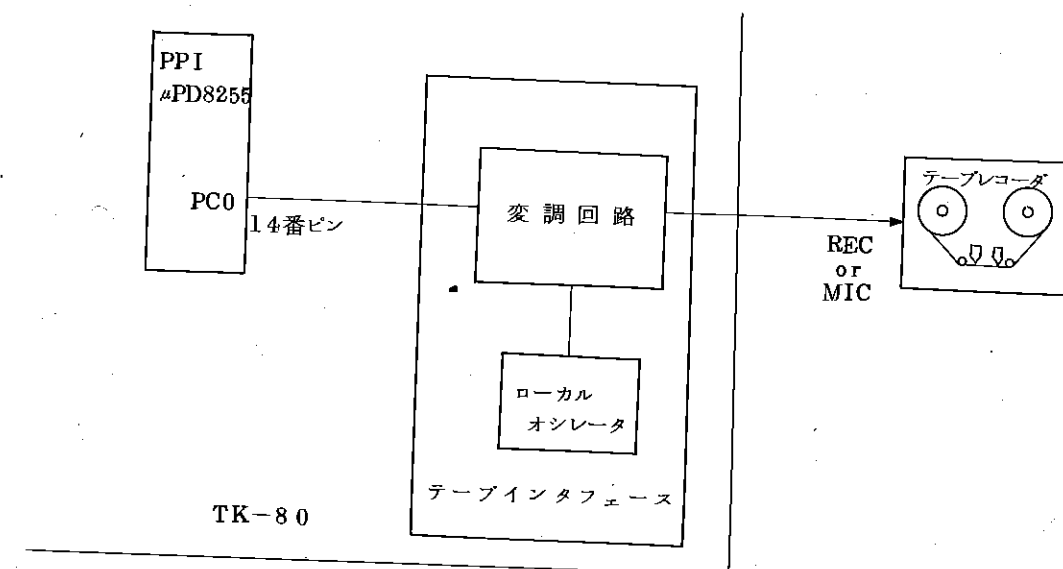
ストップビットは，3ビット長のハイレベルの信号で，1ワードのデータ転送が終了したことを示すと同時に次に転送されてくるデータとの区切りとして使用されます。

上記の例は，23（16進）というデータを転送する場合の波形です。

転送速度は，テレタイプとの互換性を考えテレタイプと同じにしてあります。

ストップビットが，3ビットあるために1ワードが12ビット構成となりますので，1ワードの転送に必要な時間は約110msecということになり，256ワードの転送は約28秒，1Kワードの転送は約2分ということになります。

6.3 データの送信

図6－2　データ送信インタフェース

送信用データは，モニタプログラムによりPPI（μPD8255）のPC0端子（14番ピン）に出力されます。

モニタプログラムは，セットされた転送スタート番地とエンド番地をパラメータとしてその範囲のメモリの内容を1つのブロックとして転送します。

この時のデータブロックは次のように構成されます。

スタート番地〔HI〕	スタート番地〔LO〕	エンド番地〔HI〕	エンド番地〔LO〕	データ	データ	データ	――――――――	データ	チェックサムコード

ここで各データ（8ビット）は，6.2項のフォーマットに従って構成されています。

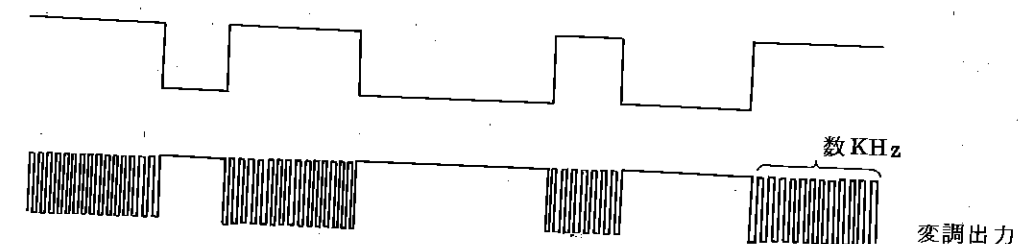
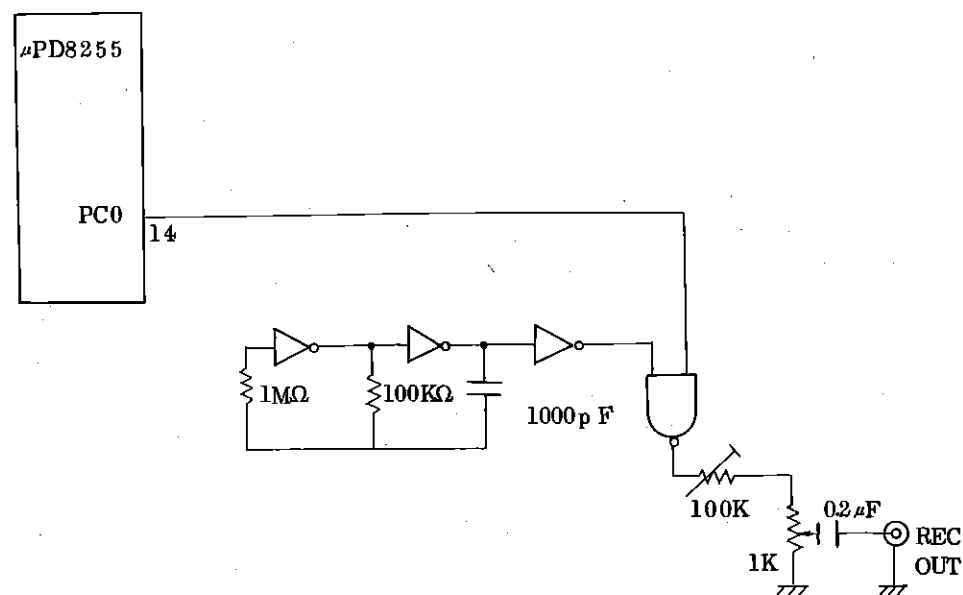
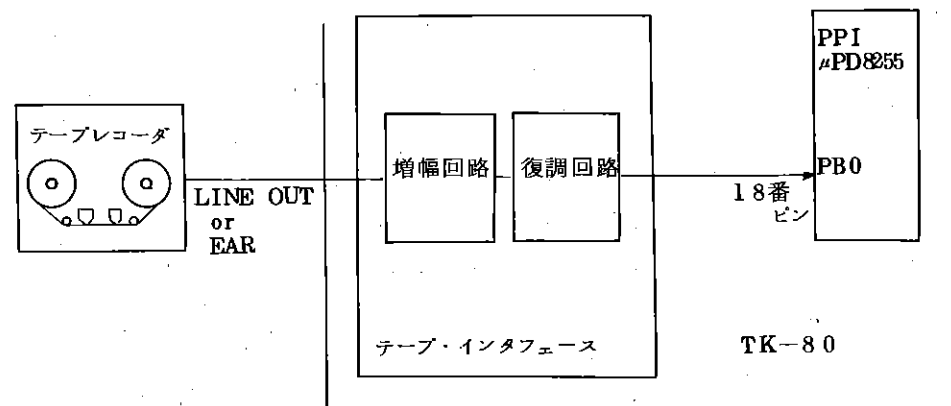
PPIのポートCより出力された転送データは，カセットインタフェースの変調回路により，次のような信号に変調されます。

6.4　変調回路

図6－3　変調回路

変調回路は，C－MOSのインバータによって構成されている発振器（発振周波数数KHz）によって発振させたパルスを出力ポートからの転送データ信号によってスイッチングするという簡単なものです．

6.5　データの受信

図6－4　データ受信インタフェース

テープレコーダからの再生出力は，テープ・インタフェース内の増幅回路で増幅され復調回路において再びデジタル信号に復調されます．

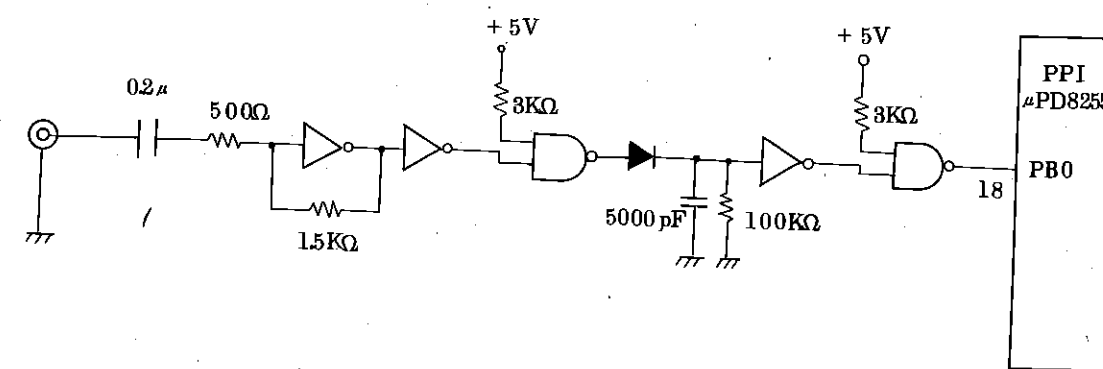
モニタプログラムは［LOAD DATA］キーが押されると，ポートBのビット0（PB0）をセンスしはじめます．

ここでスタートビット（ロウレベル）が検出され，そのビットの中央でスタートビットが確認されると，1ビット分の時間のインターバルをおきながらデータを読み込み，シリアルデータを8ビットのデータに編集していきます．

モニタは，転送されてくるデータの最初の4ワードで転送スタート番地とエンド番地を受け取り，順次アドレスを更新しながら所定の番地にデータをロードしていきます．

データのロードが終了すると，最後に転送されてくるチェック・サム・コードを受け取り，今までに受信したデータにエラーがないかどうかをチェックします．

6.6　復調回路

図6－5　復調回路

テープレコーダの復調出力は，C－MOSのインバータによって構成された増幅器によりロジックレベルまで増幅された後，ダイオードとCRで構成された積分回路により復調されバッファを介して入力ポートにデジタル信号として供給されます．

6.7　インタフェース製作および使用法

6.7.1　部　品　表

以下の部品を用意してください.

注　これらの部品はキット内には含まれておりません.

I C	μPD4049C	又は相当品	1
	μPB201C／7400	又は相当品	1
ダイオード	1S953		1
R	1MΩ　1／4W		1
	100KΩ　1／4W		2
	3KΩ　1／4W		2
	1.5KΩ　1／4W		1
	500Ω　1／4W		1
	100KΩ	半固定抵抗	1
	1KΩ	半固定抵抗	1
C	1,000 pF	25V程度　セミラック，マイラ	1
	10,000 pF	25V程度　セミラック，マイラ	1
	0.2 μF	25V程度　セミラック，マイラ	2

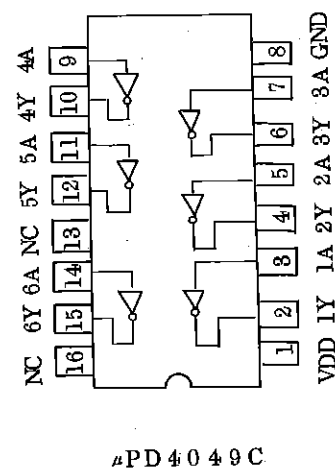
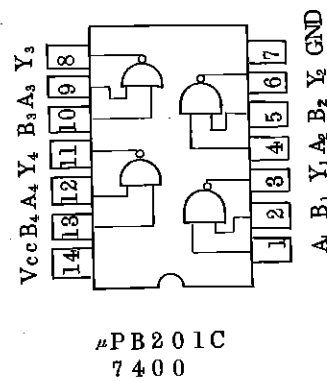
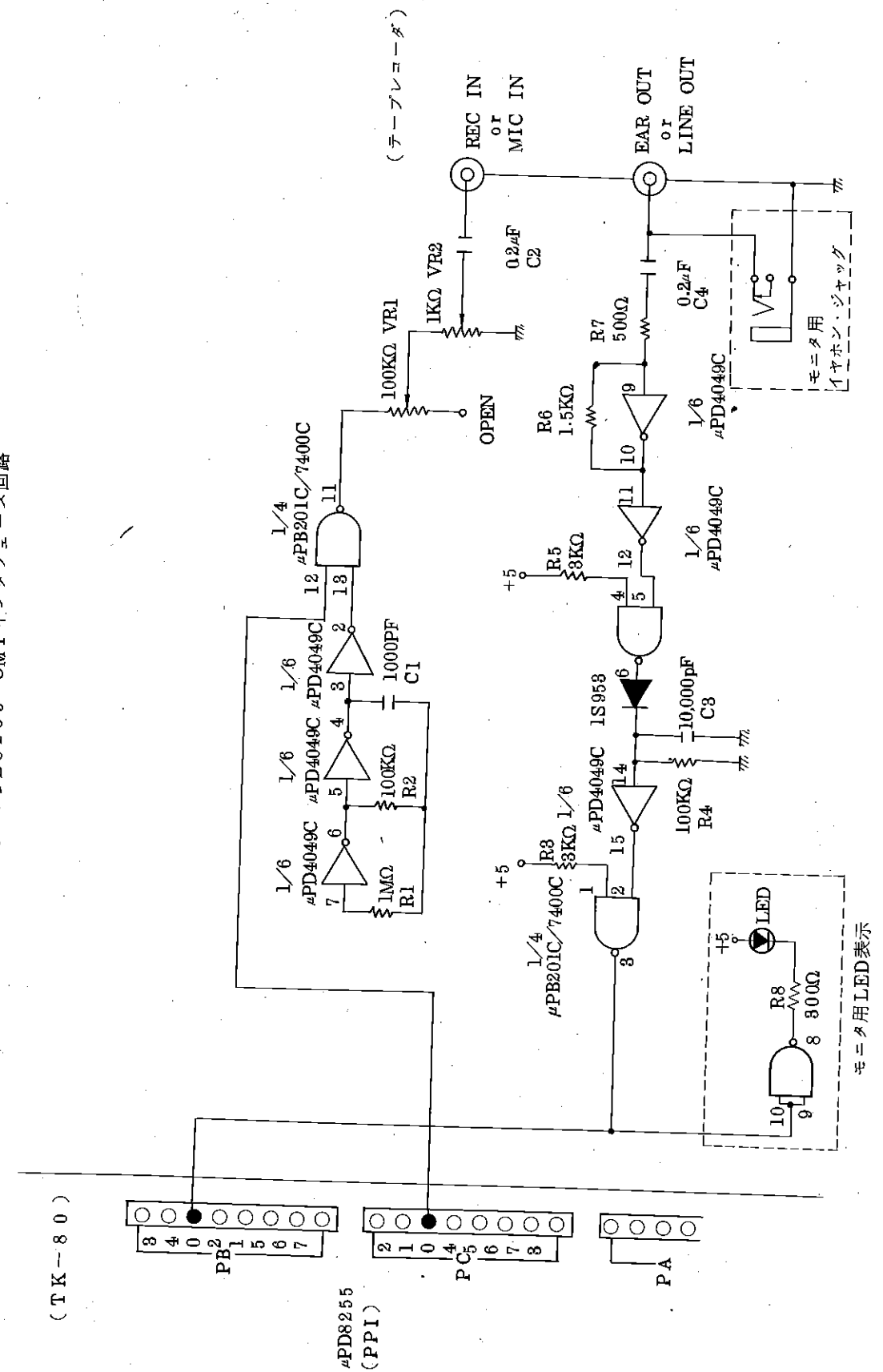



図6－6　μPD8255－CMTインタフェース回路

6．7．2　テープへの録音

変調回路の出力をマイク入力端子かライン入力端子に接続します．この時使用するケーブルは，外部ノイズを考慮し，シールド線を使用してください．

これでデータ転送の準備ができたわけですが，実際にデータを録音する前に録音入力レベルを適正なものに調整する必要があります．

録音レベルは，半固定抵抗ＶＲ１およびＶＲ２で行います．

[RESET] キーを押すと，テープ録音端子からは，インタフェース内にあるローカルオシレータの発振信号（数ＫＨｚ）が出てきます．

この信号を録音して，録音状態を調べ最適な録音レベルとなるように，ＶＲ１およびＶＲ２によって録音出力レベルを調整します．

もしテープレコーダにＶＵメータ（録音レベルメータ）が付いているならば，ＶＵメータを見ながら録音レベルの調整を行ってください．

調整は，まずＶＲ２を中間位置にセットして，ＶＲ１を調整します．その後ＶＲ２で微調整を行います．

録音レベルが適当でない場合は，テープからプログラムをロードする際のエラー原因となります．

録音レベルの調整が終了しますといよいよプログラムを転送，録音します．

まずデータキーにより，転送開始番地をデータレジスタにセットし [ADRS SET] キーを押します．続いて，転送終了番地をデータレジスタにセットします．

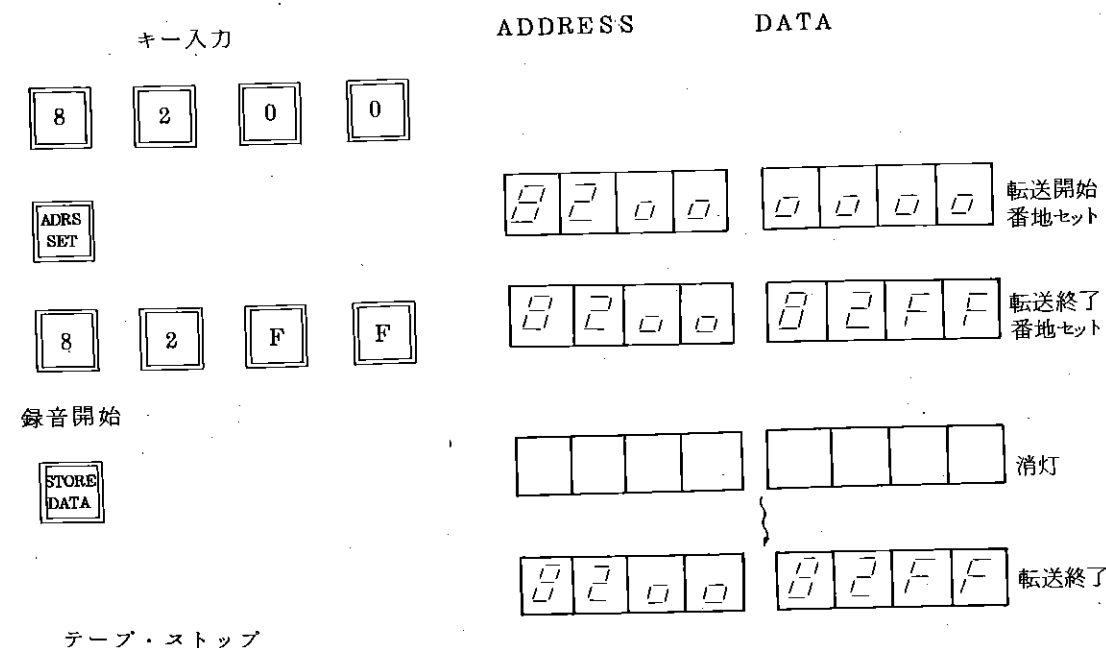
ここでテープレコーダを録音状態として，先ほどの発信を数十秒録音します．

この後 [STORE DATA] キーを押してください．

[STORE DATA] キーが押されるとＬＥＤディスプレイ上の表示が消灯し，データの転送がはじまります．

データの転送が終了すると，再びＬＥＤディスプレイが点灯して，データの転送が終了したことを表します．

これを確認した上でテープレコーダをストップさせてください．

キー入力	ADDRESS	DATA	
[8] [2] [0] [0]			
[ADRS SET]	8200	0000	転送開始番地セット
[8] [2] [F] [F]	8200	82FF	転送終了番地セット
録音開始			
[STORE DATA]	（消灯）	（消灯）	消灯
	8200	82FF	転送終了
テープ・ストップ			

6．7．3　データのロード

テープレコーダのイヤホーン出力端子か，ライン出力端子を復調回路の入力端子に接続します．

接続ケーブルは，シールド線が望ましいでしょう．

テープよりデータをロードする場合も，テープレコーダの出力レベルの調整が必要です．

復調回路の入力レベルは，１Ｖ程度必要です．ポータブル・テープレコーダのイヤホーン出力端子は，スピーカと並列になっているものが多いためこの程度の出力レベルが得られますが，もしこの出力レベルが得られない場合は，復調回路の前段で電力増幅を行ってください．また出力信号が歪んでいるとエラーの原因になりますので，出力信号が歪んでいないことを確認してください．

調整が終了すると，データのロード操作にうつります．

回路図に付加されているモニタ用イヤホーンジャックがついている場合は，イヤホーンでテープ出力をモニタしながら操作するとよいでしょう．

[RESET] キーを押しモニタプログラムを走らせます．

次にテープレコーダの再生を開始し，データの前に録音されているマーク音（4KHz程度の発振音でデータ転送を開始する前に録音したもの）を確認した後 [LOAD DATA] キーを押します．

この操作によりＬＥＤディスプレイが消灯し，データの受信が開始されます．

テープには，データのロード先頭番地とエンド番地も記録されているため，データは自動的にその領域にロードされます．

データのロードが終了するとサムチェックを行い，受信したデータにエラーがないことを自動的に確認します．

ここでエラーがないことが確認されると，アドレス・ディスプレイにデータ転送のスタート番地，データ・ディスプレイにはエンド番地が表示されます．

もしエラーが検出されますと，ＬＥＤディスプレイにエラーメッセージが表示されます．

キー入力　　ADDRESS　　DATA

[RESET]

テープ再生開始

マーク音確認

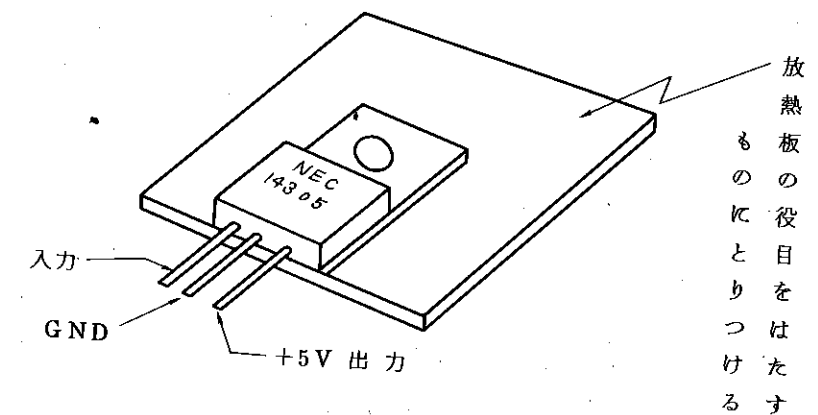
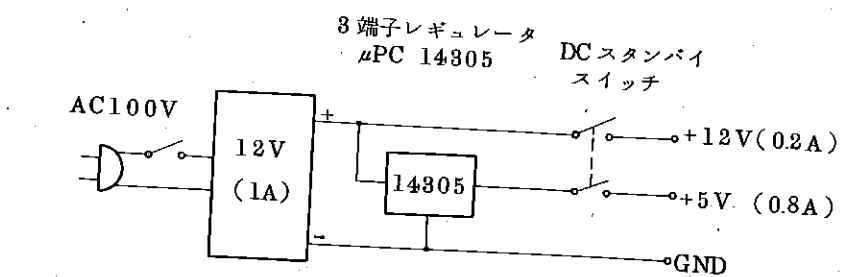
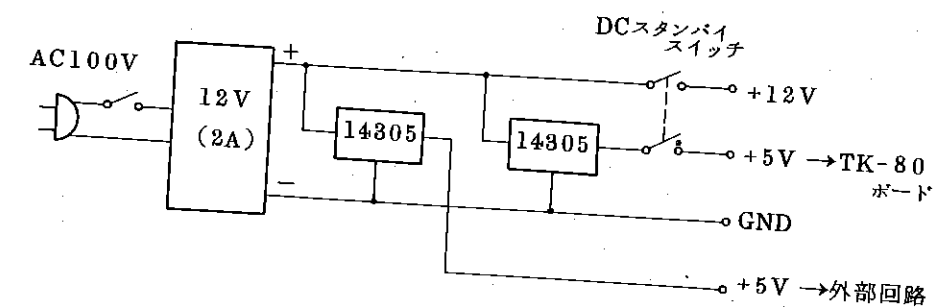
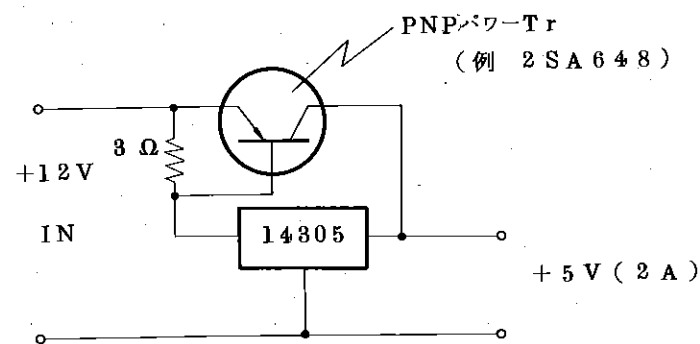
受信終了エラーあり

また最初に入っている転送スタート番地もしくはエンド番地をリードエラーしますと，前記のような表示はされず，受信状態から抜け出してこないことがあります.

エラーが発生した場合，その原因は単発的なノイズか録再生レベルが不適当であることが考えられますので，まずは再生レベルを再調整してやりなおしてください. また前記のように受信状態から抜け出してこない場合は，［RESET］キーを押してからやりなおしてください.

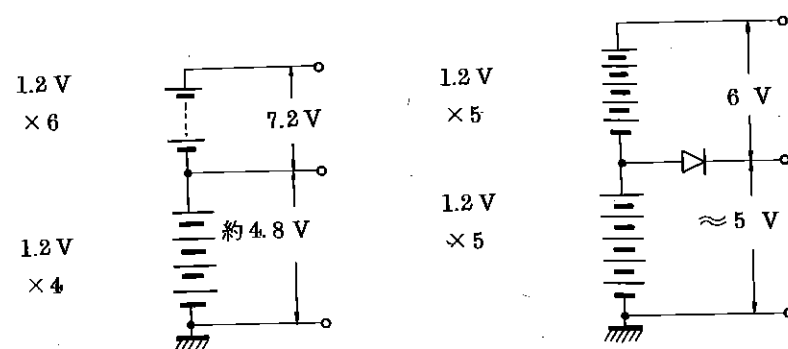
第7章　TK-80用電源回路例

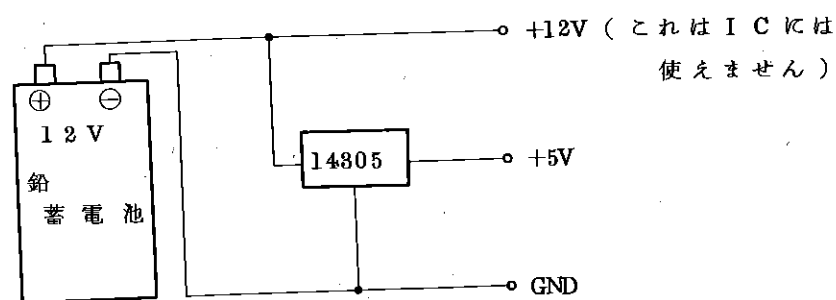
7.1　3端子レギュレータを使用する場合

TK－80には，＋12V，＋5Vの2電源が必要ですが＋12V電源の電流容量に余裕があれば，＋12Vから＋5Vを作り出して使用することができます.

＋5Vの電流容量をさらにふやしたい場合には，次の図のように3端子レギュレータを複数個使うとよいでしょう.

3端子レギュレータ1個で済ませたい場合には，次のような回路でも電流容量は増やせます. ただし，レギュレーションは多少悪くなります.

7.2 バッテリー動作

ニッケル・カドミウムの充電可能な電池を使うのが便利です.

この場合端子電圧は，1個当たり1.2V(公称電圧)ですので，+5Vとしては4個直列の4.8Vか，5個直列の6Vをダイオードの順方向電圧分落として5V近くで使用できます.

ニッケル・カドミウム電池の場合

容量の大きい12Vの鉛蓄電池がある場合，3端子レギュレータで5Vに落とすようにしておけば，電池の端子電圧が相当減少しても使用できます.

12V鉛蓄電池の場合は簡単です.

注 自動車のバッテリーから電源を供給する場合エンジンの回転数に応じてバッテリーの端子電圧が変動しますので，直接TK−80の+12V電源には接続しないでください.

付図．Ｉ　プリント基板端子配列表

端子番号	端　子　名	端子番号	端　子　名
A　1	GND	B　1	GND
2	GND	2	GND
3	+5V	3	+5V
4		4	
5	+12V	5	+12V
6		6	
7		7	
8		8	
9		9	
10	AB15	10	AB　7
11	AB14	11	AB　6
12	AB13	12	AB　5
13	AB12	13	AB　4
14	AB11	14	AB　3
15	AB10	15	AB　2
16	AB　9	16	AB　1
17	AB　8	17	AB　0
18		18	
19		19	
20		20	
21		21	
22		22	
23		23	
24		24	
25		25	
26		26	DB　7
27		27	DB　6
28		28	DB　5
29		29	DB　4
30		30	DB　3
31		31	DB　2
32		32	DB　1
33		33	DB　0
34		34	
35		35	
36		36	
37		37	
38		38	
39		39	
40		40	
41		41	
42		42	
43		43	
44		44	
45		45	
46		46	
47		47	
48		48	
49		49	
50	GND	50	GND

付図III　プリント基板部品面図

8080 A
13
8228
14
8224
8255
15
8212
28
2155
555
29
454-0
454-1
454-2
454-3
5101
KEY SWITCH
TAPE LOAD
TAPE STOR
MEM WRIT
ADRS SET
RET
RUN
713A
1 74LS04 (14PIN)
2 201 (〃)
3 214 (14PIN)
4 202 (14PIN)
5 215 (14PIN)
6 223 (14PIN)
7 2155 (16PIN)
8 2155 (16PIN)
9 74LS04 (14PIN)
10 238 (14PIN)
11 238 (14PIN)
12 8224 (16PIN)
13 8080A (40PIN)
14 8228 (28PIN)
15 8255 (40PIN)
16 454 (24PIN)
17 454 (24PIN)
18 454 (24PIN)
19 454 (24PIN)
20~27 5101 (22PIN)
28 8212 (24PIN)
29 555 (8PIN)
版
記事
承認
発行
年月日
特に指定のないものは普通公差
第三角法
尺度
設計
査閲
承認
T. Goto
76. 6. 10
仕様書
組立図
部品表
素材名
所要寸度
仕上および処理
付図Ⅱ　TK－80回路図
フイルム
AP ROLL
整理期限
整理番号
D201-0008
JIS A2 (420×594)
NEC 日本電気株式会社